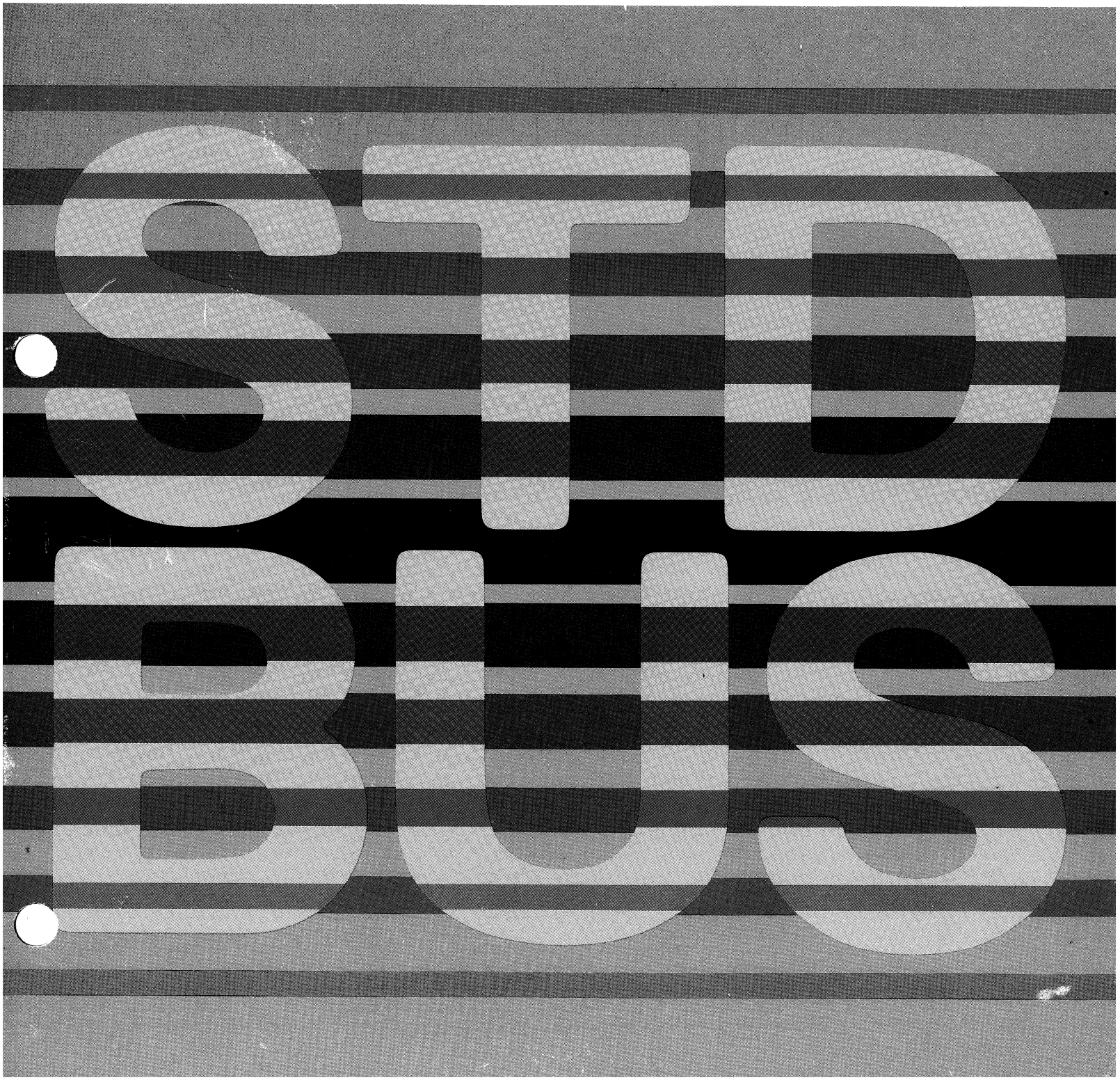
PRO-LOG®
CORPORATION

STD 7000

7863-1/7863-3 MULTIFUNCTION 8088 CPU/COMMUNICATION CARDS

USER'S MANUAL



NOTICE

The information in this document is provided for reference only. Pro-Log does not assume any liability arising out of the application or use of the information or products described herein.

This document may contain reference information and products protected by copyrights or patents and does not convey any license under the patent rights of Pro-Log, nor the rights of others.

Printed in U.S.A. Copyright 1985 by Pro-Log Corporation, Monterey, California 93940. All rights reserved.

PRO-LOG®
CORPORATION

STD 7000

7863-1/7863-3
MULTIFUNCTION 8088
CPU/COMMUNICATION CARDS

USER'S MANUAL

7/86

FOREWORD

This manual explains how to use the Pro-Log 7863 Multifunction 8088 CPU/Communication Card. It is structured to reflect the answers to basic questions about the 7863. We welcome your suggestions on how we can improve our instructions.

The 7863 is part of the Pro-Log Series 7000 STD BUS hardware. Our products are modular and provide managers with the means of using their own people to control the design, production, and maintenance of products that use STD BUS hardware.

Our products are designed to quality standards with readily available, second-sourced, industry standard components, clear and complete specifications, full documentation, and the option to purchase the manufacturing rights and drawings.

You may also find the *STD BUS Technical Manual and Data Book* useful in your work. If you would like a copy of this document, please write to us on your company letterhead.

CONTENTS

Page

Foreword ii
Figures v

Section 1. Specifications 1-1

Purpose 1-1
Features 1-1
Functional Description 1-4
Specifications 1-6
Electrical and Environmental Specifications 1-6
Mechanical Specifications 1-6
Timing Specifications 1-8

Section 2. 7863-1 Installation (Multimaster) 2-1

Introduction 2-1
Power Supply 2-1
Physical Installation 2-1
Priority Chain 2-2
Card Layout and Jumpers 2-4
Multimaster Configuration 2-6
Memory and I/O 2-8
Interrupt Options 2-10
RS-422-A Serial Communications Channel 2-11
RS-232-C Serial Communications Channel 2-12

Section 3. 7863-3 Installation (Single Master) 3-1

Introduction 3-1
Power Supply 3-1
Physical Installation 3-1
Priority Chain 3-2
Card Layout and Jumpers 3-2
Reset, Jumper W5 (Figure 2-5) 3-5
Memory and I/O 3-5
Interrupt Options 3-7
RS-422-A Serial Communications Channel 3-8
RS-232-C Serial Communications Channel 3-9

Section 4. 7863-1 Operation and Programming 4-1

Introduction 4-1
Single-CPU Systems 4-1
Multimaster STD BUS Systems 4-1
Bus Arbitration 4-1
Bus Lock 4-2
Multimaster System Configurations 4-2
Bus Protocol 4-3
Share Memory (Mailboxes) 4-3
Software Interrupts 4-4
Periodic STD BUS Access 4-4
Interrupts 4-5

Operation and Programming 4-6
Reset 4-6
Interrupts 4-7
Memory Addressing 4-8
I/O Addressing 4-11

Section 5. 7863-3 Operation and Programming 5-1

Introduction 5-1
Operation and Programming 5-1
Reset 5-1
Interrupts 5-1
Memory Addressing 5-3
I/O Addressing 5-3

Section 6. Serial Communications Controller (SCC) 6-1

Introduction 6-1
Internal Structure 6-2
SCC Receive and Transmit Registers 6-6
SCC Control Registers 6-6
SCC Status Registers 6-21
SCC Operation 6-29
Initialization 6-29
Baud Rate 6-30
Interrupts 6-30
RS-232-C Handshaking (Channel A) 6-31
Block Transfers 6-31

Section 7. Operating Software 7-1

Introduction 7-1
Description 7-1
User Requirements 7-1
Flowcharts 7-3
MACRO Assembler Listing 7-7

Section 8. Maintenance 8-1

Reference Drawings 8-1
Return for Repair/Warranty 8-7

FIGURES

Figure		Page
1-1	7863-1 Multifunction 8088 CPU Communication Card	1-2
1-2	7863-1 Block Diagram	1-2
1-3	7863-3 Multifunction 8088 CPU Communication Card	1-3
1-4	7863-3 Block Diagram	1-3
1-5	Onboard Memory Mapping Options	1-5
1-6	I/O Port Address Assignments	1-5
1-7	Recommended Memory Devices	1-6
1-8	STD Edge Connector	1-7
1-9	RS-422-A Serial Connector J2	1-7
1-10	RS-232-C Serial Connector J1	1-8
1-11	Bus Status Timing	1-9
1-12	Bus Read Timing	1-10
1-13	Bus Write Timing	1-11
1-14	Bus Interrupt Timing	1-12
1-15	Bus Wait Request Timing	1-13
1-16	Bus Exchange Timing	1-14
1-17	Multimaster Timing	1-15
2-1	Installation	2-1
2-2	Multimaster Installation Options	2-3
2-3	Jumper Configuration	2-4
2-4	Sources for Shorting Jumpers	2-4
2-5	Physical Layout and Jumper Options	2-4
2-6	Multimaster Configuration Jumpers	2-7
2-7	CNTRL* Source Select, Jumper W2	2-7
2-8	PCO Drive Enable, Jumper W10	2-7
2-9	Lock Mode Select, Jumper W10	2-8
2-10	Reset Voltage Select, Jumper W5	2-8
2-11	Recommended Memory Devices	2-9
2-12	Memory Socket Configuration Jumpers W8 (MEM1) and W9 (MEM0)	2-9
2-13	Onboard Memory Mapping, Jumper W11	2-10
2-14	IOEXP Drive Example, Jumper W1	2-10
2-15	INTRQ* Enable, Jumper W3	2-11
2-16	INTRQ* Vector Source Select, Jumper 12	2-11
2-17	RS-422-A Synchronous System Interconnection	2-11
2-18	RS-422-A Configuration, Jumper W15	2-12
2-19	RS-422-A Receive Clock Source, Jumper W16	2-12
2-20	RS-422-A TCLK/RTS Select, Jumper W14	2-12
2-21	RS-232-C Configuration, Jumpers W13 and W18	2-12
2-22	RS-232-C Serial Port DCE/DTE and DTE Clock Jumpers	2-13
2-23	RS-232-C External DTE Clock, Jumper W17	2-13
3-1	Installation	3-2
3-2	Jumper Configuration	3-2
3-3	Sources for Shorting Jumpers	3-3
3-4	Physical Layout and Jumper Options	3-3
3-5	Reset Voltage Select, Jumper W5	3-5
3-6	Recommended Memory Devices	3-6
3-7	Memory Socket Configuration Jumpers W8 (MEM1) and W9 (MEM0)	3-6

3-8 Onboard Memory Mapping, Jumper W11 3-7
3-9 IOEXP Drive Example, Jumper W1 3-7
3-10 INTRQ* Enable, Jumper W3 3-7
3-11 INTRQ* Vector Source Select, Jumper W12 3-8
3-12 RS-422-A Synchronous System Interconnection 3-8
3-13 RS-422-A Configuration, Jumper W15 3-8
3-14 RS-422-A Receive Clock Source, Jumper W16 3-9
3-15 RS-422-A TCLK/RTS Select, Jumper W14 3-9
3-16 RS-232-C Configuration, Jumpers W13 and W18 3-9
3-17 RS-232-C Serial Port DCE/DTE and DTE Clock Jumpers 3-10
3-18 RS-232-C External DTE Clock, Jumper W17 3-10

4-1 Use of XCHG Instruction with Lock Prefix 4-3
4-2 Multimaster Swing Buffer 4-4
4-3 Multimaster Communication with Software Interrupts 4-5
4-4 Communications with Periodic Interrupts 4-6
4-5 NMIRQ* Processing 4-7
4-6 INTRQ* Processing 4-8
4-7 Memory Mapping Examples 4-9
4-8 I/O Port Configuration Examples (8-, 10-, and 16-Bit) 4-11

5-1 NMIRQ* Processing 5-2
5-2 INTRQ* Processing 5-2
5-3 Memory Mapping Examples 5-3
5-4 I/O Port Configuration Examples (8-, 10-, and 16-Bit) 5-4

6-1 Examples of Various Formats 6-1
6-2 Data Signaling Rate vs Cable Length for Balanced Interface 6-2
6-3 SCC Register Structure 6-2
6-4 Write Register Bit Maps 6-3
6-5 Read Register Bit Maps 6-5
6-6 SCC Control Register Functions 6-6
6-7 Write Register 0 Bit Map 6-7
6-8 Write Register 1 Bit Map 6-8
6-9 Write Register 2 Bit Map 6-10
6-10 Write Register 3 Bit Map 6-10
6-11 Write Register 4 Bit Map 6-11
6-12 Write Register 5 Bit Map 6-12
6-13 Write Register 6 Bit Map 6-13
6-14 Write Register 7 Bit Map 6-13
6-15 Write Register 9 Bit Map 6-14
6-16 Write Register 10 Bit Map 6-15
6-17 Write Register 11 Bit Map 6-17
6-18 Write Register 12 Bit Map 6-17
6-19 Write Register 13 Bit Map 6-18
6-20 Write Register 14 Bit Map 6-18
6-21 Write Register 15 Bit Map 6-20
6-22 Read Register Functions 6-21
6-23 Read Register 0 Bit Map 6-24
6-24 Read Register 1 Bit Map 6-25
6-25 Read Register 2 Bit Map 6-26
6-26 Status Vector Mode Table 6-27

6-27 Read Register 3 Bit Map 6-27
6-28 Read Register 10 Bit Map 6-27
6-29 Read Register 12 Bit Map 6-28
6-30 Read Register 13 Bit Map 6-28
6-31 Read Register 15 Bit Map 6-29
6-32 SCC Initialization Order 6-29
6-33 Time Constants for Common Baud Rates 6-30
6-34 SCC Internal Interrupt Priority 6-30
6-35 SCC RS-232-C Handshake Capability (Channel A) 6-31

7-1 Subroutine Types 7-1
7-2 Main Subroutines Use Requirements 7-2
7-3 (INIT COM) Flowchart 7-3
7-4 (GEN INT TABLE) Flowchart 7-3
7-5 (INIT SCCA)/(INIT SCC B) Flowchart 7-3
7-6 (XMT CH A ASYN) Flowchart 7-4
7-7 (XMT CH B SDLC) Flowchart 7-4
7-8 (XMT CHAR) Flowchart 7-4
7-9 (END XMT?) Flowchart 7-4
7-10 (RCV CHAR A) Flowchart 7-5
7-11 (RCV CHAR B) Flowchart 7-5
7-12 (SPCL RCV A) Flowchart 7-6
7-13 (SPCL RCV B) Flowchart 7-6
7-14 Program Listing 7-7

8-1 Schematic 8-2
8-2 Assembly Drawing 8-5

Section 1
SPECIFICATIONS

PURPOSE

The 7863 is a multifunction 8088 CPU card with up to 128K of onboard memory, two serial communication ports, and an optional 8087 math coprocessor. This card is available in two versions:

1. The **7863-1** is a multimaster CPU card.

2. The **7863-3** is an 8088 CPU card used as a standard single processor.

The multimaster version, 7863-1, has specially designed bus arbitration logic that allows the use of more than one CPU (7863-1 or other Pro-Log multimaster CPU cards) in one STD BUS backplane, each with access to global memory and I/O (i.e., memory and I/O not located on a multimaster CPU card). Using these CPU cards, a variety of multiprocessor system configurations can be implemented on the STD BUS.

Bus arbitration is done entirely in hardware, making the arbitration process transparent to software on any of the multimaster CPU cards. Wait states are automatically inserted when the 7863 tries to access global (off-card) memory or I/O while the bus is being used by another multimaster or during the actual time required for bus arbitration. A multimaster always relinquishes the bus at the end of any machine cycle during which another multimaster has requested the bus unless it has "locked" the bus with either software (8088 lock instruction prefix) or hardware (onboard lock control port). The STD BUS priority chain (PCI/PCO) is used to resolve and prioritize simultaneous bus access requests from more than one multimaster CPU card.

Both multimaster and single master (7863-3) versions feature a buffered and fully expandable 8088 microprocessor operated at 4.9152 MHz in the maximum mode. The optional 8087 math coprocessor greatly simplifies and speeds up arithmetic calculations, while for software purposes it can be treated as an extension of the 8088's registers and instruction set.

The two onboard serial communication ports are configured as one RS-422-A and one RS-232-C channel. Either channel can be used synchronously or asynchronously. In synchronous operation, BISYNC, HDLC, SDLC, and self-clocking (FM and NRZI) modes are supported. The serial communications controller (SCC) is compatible with the 8088 vectored interrupt scheme and can provide a variable vector for fast response to different interrupting conditions.

The two memory sockets conform to the JEDEC 28-pin standard and either accepts 8K, 16K, 32K, or 64K ROMs; or 8K or 32K byte-wide RAMs. Eight mapping options are provided for each socket. Usually one socket is mapped at the high end of the one megabyte 8088 memory map to include the 8088's power-on restart address, while the other socket is mapped at the low end to support the 8088 interrupt structure. This onboard memory is available exclusively to the 8088 CPU on the 7863; it cannot be accessed by other multimasters in the system.

FEATURES

* 8088 CPU with optional 8087 math coprocessor

* Multimaster capability on the 7863-1 (up to 16 CPU cards on one STD backplane)

* Up to 128K bytes of onboard ROM

* Up to 64K bytes of onboard RAM (32K by 8 devices)

* Direct addressing of 1M byte of memory, 32K off-card I/O ports

* Two serial communication ports:
  -one RS-422-A port and one RS-232-C port
  -asynchronous, BISYNC, HDLC, and SDLC communications
  -software baud rate selection up to 819K baud

* Precision power-on and pushbutton reset circuitry

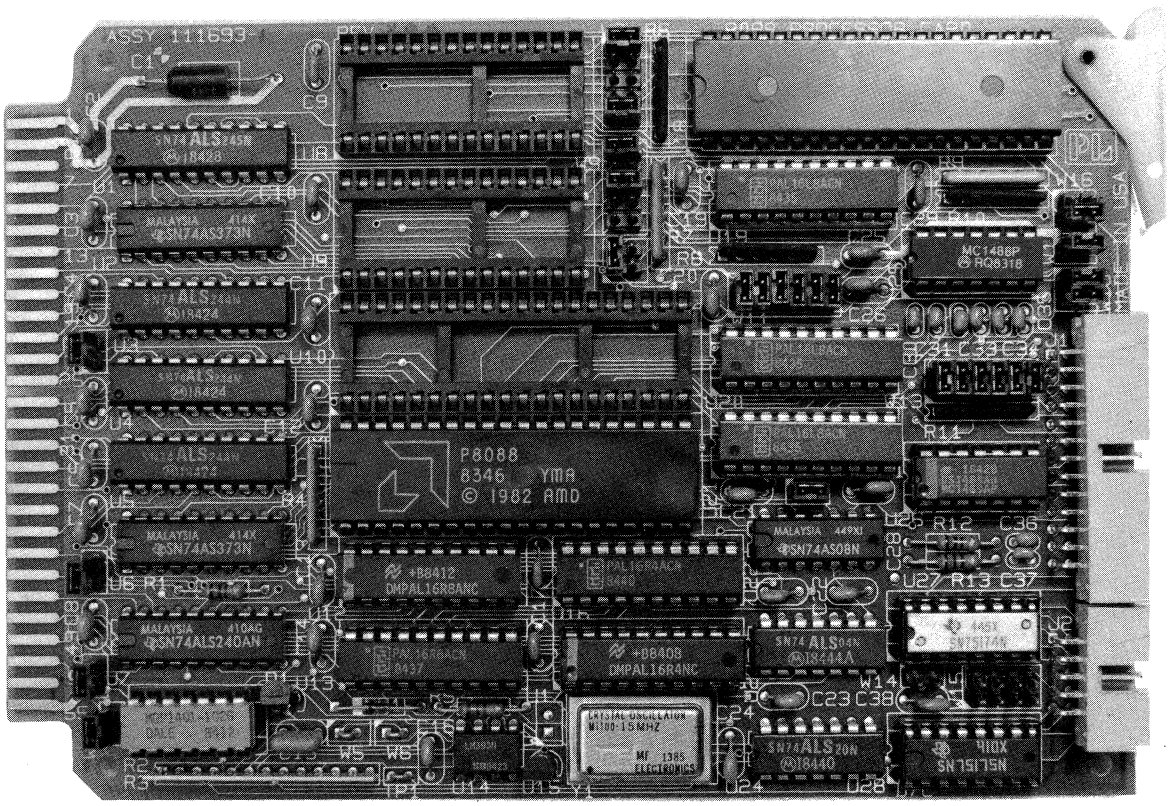


Figure 1-1. 7863-1 Multifunction 8088 CPU/Communication Card

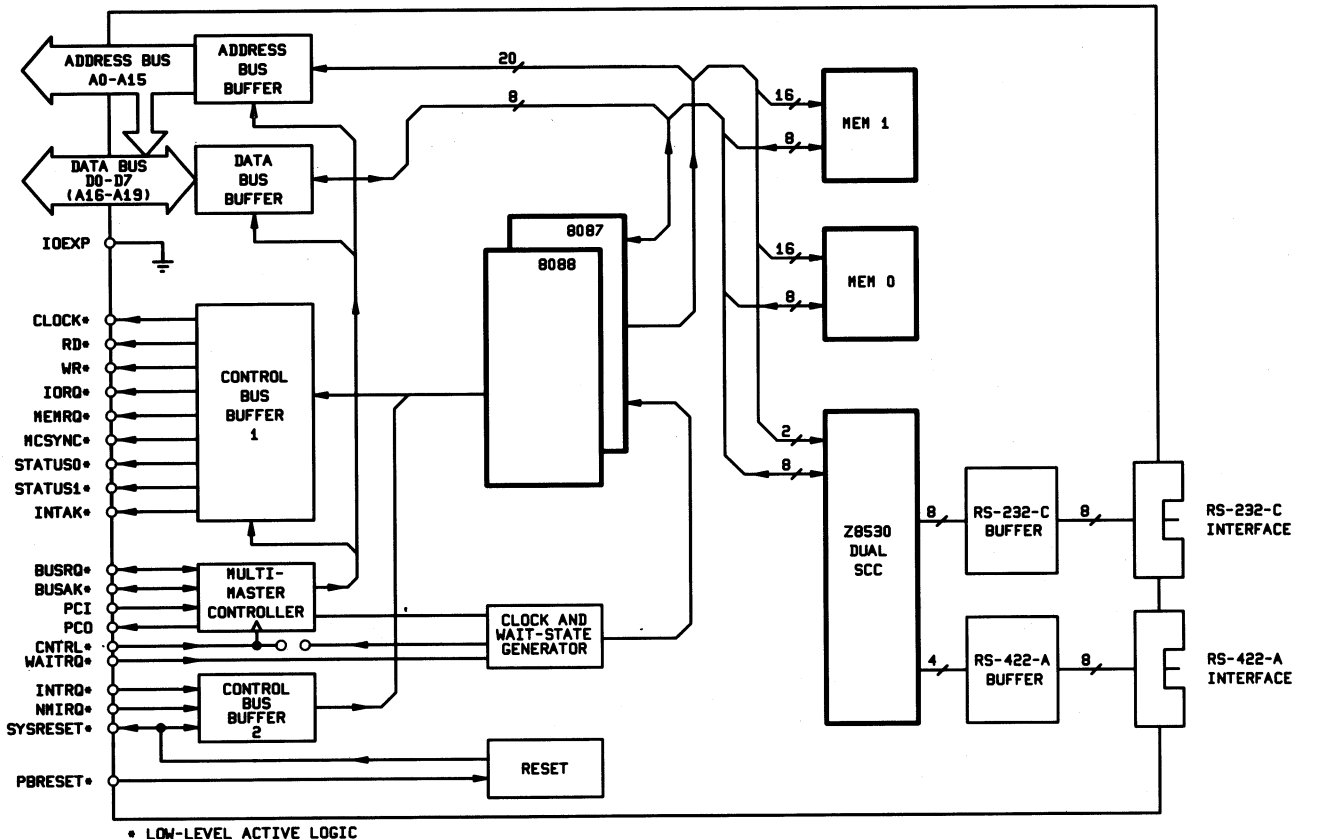


Figure 1-2. 7863-1 Block Diagram

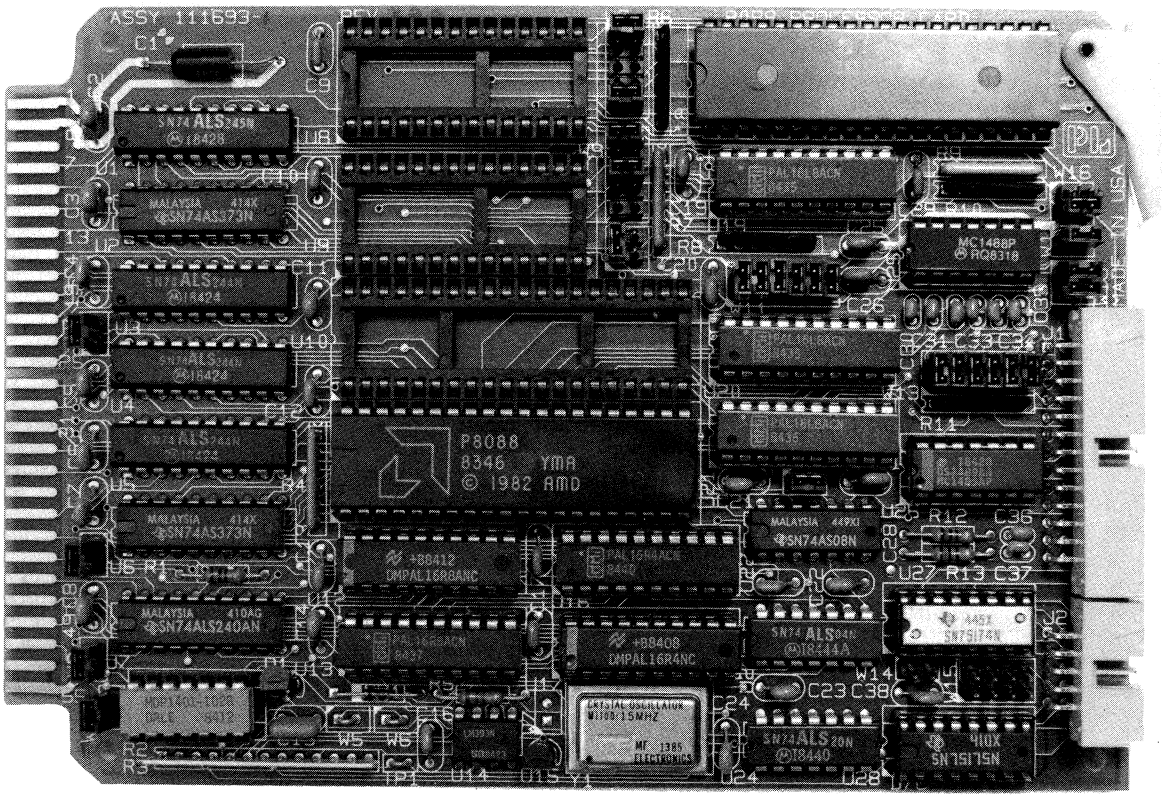


Figure 1-3. 7863-3 Multifunction 8088/CPU Communication Card

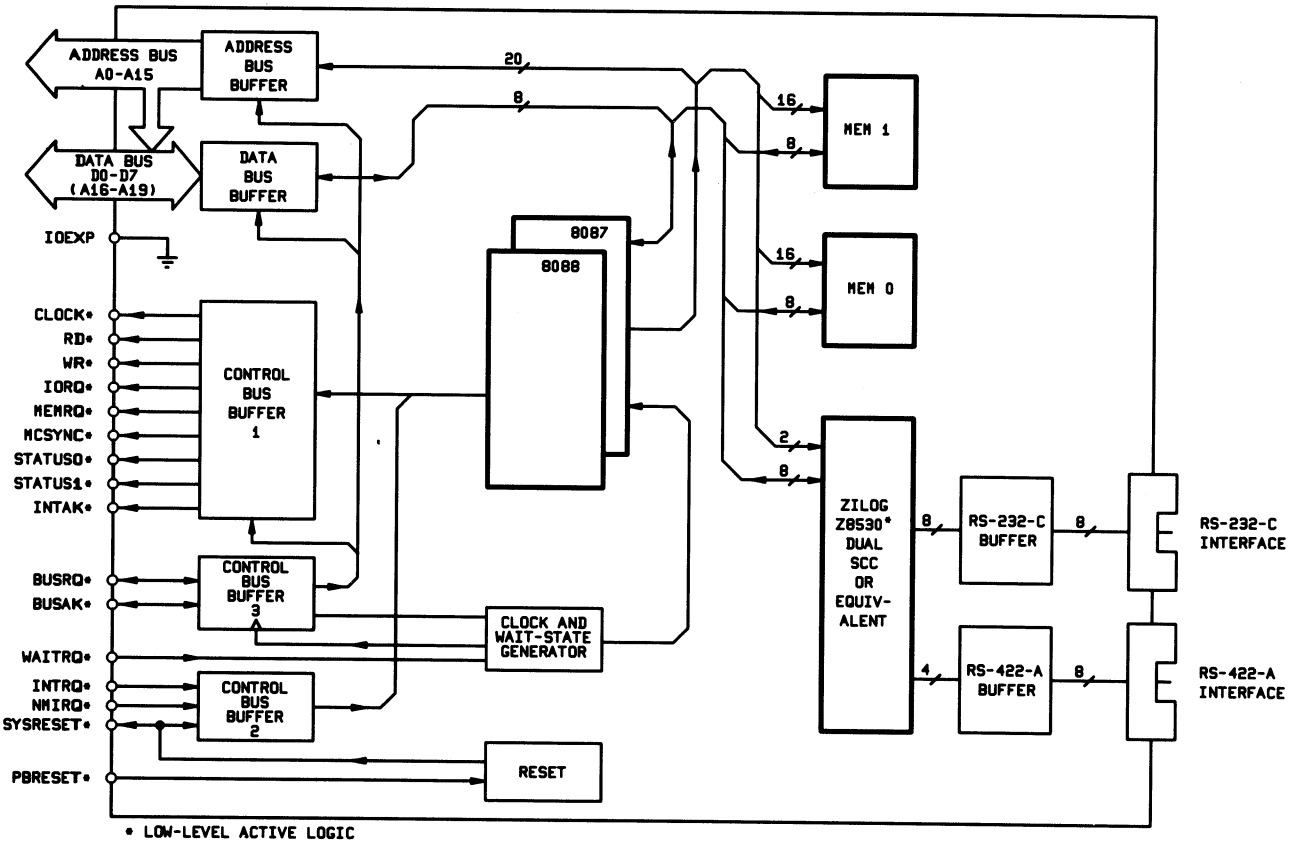


Figure 1-4. 7863-3 Block Diagram

7863-3 MULTIFUNCTION 8088 CPU/COMMUNICATION CARD
BK7865.XB1 12MAY86 PL

FUNCTIONAL DESCRIPTION

This description defines each block in the diagrams (Figures 1-2 and 1-4). The description and the block diagrams are intended to familiarize the user with both the 7863 card and the terminology used throughout this manual.

Address Bus Buffer

This buffer provides for buffering and 3-state control of the STD address bus. Address lines A0-A15 are latched, while address lines A16-A19 are time multiplexed onto data bus lines D0-D3. (The rising edge of MCSYNC* is used by external memory cards to latch these upper address lines.) The address bus buffer is 3-stated by the bus arbiter when another multimaster CPU card in the system is using the bus.

Data Bus Buffer

The data bus buffer provides 3-state control, direction control, and buffering of the STD data bus (D0-D7). As noted above, the lower four data lines are time multiplexed with the upper four address lines. Like the address bus buffer, the data bus buffer is controlled by the bus arbiter.

Control Bus Buffer 1

This buffer provides buffering and 3-state control of memory, I/O, and peripheral control signals. This buffer is also controlled by the bus arbiter.

Control Bus Buffer 2 (and Buffer 3 on the 7863-3)

This buffer is not controlled by the bus arbiter; it provides buffering for control signal inputs.

Bus Arbiter (7863-1 only)

The arbiter allows the use of the 7863-1 as a multimaster CPU with other multimasters installed in the same STD backplane. When the 7863-1 8088 CPU attempts an off-card memory or I/O operation, the bus arbiter allows the 7863-1 access to the STD BUS and resolves simultaneous bus access requirements from other multimasters.

8088/8087

The 7863 features a buffered and fully expandable 8088 microprocessor that operates at 4.9152 MHz in maximum mode. The 8088 is a 16-bit microprocessor (internally) with an 8-bit external data bus, a 20-bit memory address bus for directly addressing one megabyte of memory, and a 16-bit I/O address bus that can access up to 64K I/O ports. The 7863 can address 32K off-card I/O ports.

The 7863 accommodates an optional 8087 Numeric Processor Extension (NPX) that can be used to expand the computational capabilities of the CPU. The 8087 can perform fixed or floating-point arithmetic, multiplication, division, comparisons, as well as tangent and logarithmic functions.

The 8087 decodes its instructions in parallel with the 8088 CPU and can execute numerical instructions while the 8088 is proceeding with other instructions. For software purposes, the 8088 and 8087 can be considered as one CPU with greatly increased arithmetic capabilities.

Clock and Wait State Generator

This device generates the 4.9152 MHz clock signal for the 8088 and inserts wait states when required (by controlling the 8088 READY input). Wait states are inserted for the multimaster CPU (7863-1) when a bus access is attempted and another multimaster is currently using the bus (and during the subsequent bus arbitration cycle), during any I/O operation, and (if the 7863 has control of the STD BUS) when the STD BUS signal WAITRQ* is active. The clock/wait-state generator on one multimaster CPU card in a system also provides the clock (CNTRL* on the STD BUS) used to synchronize the bus arbiters on all multimasters.

Reset

Reset provides a signal that places the card in a reset state during power-up, pushbutton reset, or DC power fail (i.e., when VCC on-card drops below 4.65V). Precision voltage-sensing circuitry ensures that the card is held in the reset state during power-up until 60 milliseconds after VCC reaches 4.75V

minimum. The 7863 both drives and responds to SYSRESET* on the STD BUS. A jumper option is provided to lower this threshold if required.

MEM1

This 28-pin JEDEC-standard socket is for user-supplied memory devices (8K, 16K, 32K, or 64K ROMs; or 8K or 32K RAMs). The highest address in the device installed will normally be FFFFF hex. The socket can also be disabled (removed from the memory map) if required. See Figure 1-5 for memory mapping options.

MEM0

This socket is the same as MEM1, except that the lowest address in the device installed will normally be 00000 hex. See Figure 1-5 for memory mapping options.

Serial Communications

A high-performance Serial Communications Controller provides two serial communication ports, each with its own baud rate generator and phase-lock loop (for clock recovery). One port is configured for RS-232-C operation, the second dedicated to RS-422-A interface levels. An RS-232-C buffer translates TTL signals from channel A of the SCC into RS-232-C levels and vice versa. The RS-422-A buffer translates the TTL signals from channel B for RS-422-A operation.

The RS-422-A interface can be used for applications requiring long cable length, high noise immunity, or high data rates. The RS-422-A interface also conforms to multidrop specification RS-485, permitting multiple stations on a network.

Both serial channels can be used synchronously or asynchronously. In synchronous operation, BISYNC, HDLC, and SDLC modes are supported.

The serial communications controller (SCC) is compatible with the 8088 vectored-interrupt scheme and can provide a variable vector for fast response to different interrupting conditions.

OPTIONS*	MEMORY SOCKET 0	MEMORY SOCKET 1
OPTION 0	64Kx8 00000-0FFFF	64Kx8 F0000-FFFFF
OPTION 1	32Kx8 00000-07FFF	32Kx8 F8000-FFFFF
OPTION 2	16Kx8 00000-03FFF	16Kx8** FC000-FFFFF
OPTION 3	8Kx8** 00000-01FFF	8Kx8 FE000-FFFFF
OPTION 4	64Kx8 E0000-EFFFF	64Kx8 10000-1FFFF
OPTION 5	32Kx8 F0000-F7FFF	32Kx8 08000-0FFFF
OPTION 6	8Kx8 FC000-FDFFF	8Kx8 02000-03FFF
OPTION 7	SOCKET DISABLED	SOCKET DISABLED

*Options can be independently selected for each socket
**As shipped configuration

Figure 1-5. Onboard Memory Mapping

I/O PORT ADDRESS			OUTPUT	INPUT
8-BIT	10-BIT	16-BIT	FUNCTION	FUNCTION
00-FB	000-3FB	0000-7FFF	OFF-CARD I/O	
		8000-BFFB	DO NOT USE*	
		BFFC	SCC Ch. B RS-422-A Control	SCC Ch. B RS-422-A Status
		BFFD	SCC Ch. B RS-422-A XMT Data	SCC Ch. B RS-422-A RCV Data
		BFFE	SCC Ch. A RS-232-C Control	SCC Ch. A RS-232-C Status
		BFFF	SCC Ch. A RS-232-C XMT Data	SCC Ch. A RS-232-C RCV Data
		C000-FFFE	DO NOT USE*	
		FFFF	Bus Lock Control	(NOT USED)

*Due to I/O decoding, no off-card I/O can be mapped in this address block.

Figure 1-6. I/O Mapping

MEMORY TYPE	MEMORY SIZE	MANUFACTURER'S PART NUMBER	ACCESS TIME (nS)
EPROM	8K	Intel 2764	
	16K	Intel 27128 AMD 27128	
	32K	Intel 27256 AMD 27256	300 max
	64K	Intel 27512 AMD 27512	
RAM	8K	Hitachi HM6264 LP-15 Toshiba TC5565 PL-15	200 max
	32K	NEC D43256C-15 Hitachi HM62256 LP-12	

Figure 1-7. Recommended Memory Devices

SPECIFICATIONS

Electrical and Environmental Specifications

Absolute Maximum Ratings Over Operating Free Air Temperature Range (Unless Otherwise Specified.

* Supply Voltage, Vcc: 0 to 7V
* Auxiliary Supply Voltage, AUX V+: 0 to 15V
* Auxiliary Supply Voltage, AUX V-: 0 to -15V
* Operating Free-Air Temperature Range: $0^{o}C$ to $+65^{o}C$
* Storage Temperature Range: $-40^{o}C$ to $+85^{o}C$
* Noncondensing Relative Humidity: 5% to 95%

Operating Conditions	MIN	TYP	MAX	UNIT
Supply Voltage, Vcc	4.75	5.00	5.25	Volt
Supply Voltage, AUX V+	11.40	12.00	12.60	Volt
Supply Voltage, AUX V-	-11.40	-12.00	-12.60	Volt
Operating Free-Air Temperature	0	25	65	°C
Noncondensing Relative Humidity	5	—	95	%

ELECTRICAL CHARACTERISTICS

Over Operating Free-Air Temperature Range (Unless Otherwise Specified)

Parameter	TEST CONDITIONS	MIN	TYP	MAX	UNITS
Icc (Vcc)	7863, no onboard memory	—	1.4	2.0	A
	7863, no onboard memory, 8087 installed	—	1.7	2.5	A
IAUX+ (AUX V+)		—	22	33	mA
IAUX= (AUX V=)		—	-19	-30	mA

Vcc TRIGGERED RESET	MIN	TYP	MAX	UNITS
W5 shorting connector installed:				
Vcc, reset delay start (power up)	4.75	—	4.90	V
Vcc, SYSRESET* low (power down)	4.55	—	4.70	V
W5 shorting connector removed:				
Vcc, reset delay start (power up)	4.60	—	4.75	V
Vcc, SYSRESET* low (power down)	4.40	—	4.55	V
Hysteresis	—	200	—	mV
Delay Reset off to SYSRESET* high	60	—	100	ms

All logic signal levels related to the 7863 that appear on the STD BUS conform to TTL requirements specified in the general *STD Series 7000 and 70C00 Specification*, Pro-Log document number 111684. See the 7863 Edge Connector Pin List for bus input loading and output drive.

7863 MTBF is over 19 years. MTBF is calculated using component standard failure rates specified in *Failure Rates and MTBF Calculation*, Pro-Log document number 111443.

PIN NUMBER	OUTPUT (LSTTL DRIVE)	INPUT (LSTTL LOADS)		PIN NUMBER	OUTPUT (LSTTL DRIVE)	INPUT (LSTTL LOADS)	
MNEMONIC							MNEMONIC
+5V	IN		2	1		IN	+5V
GROUND	IN		4	3		IN	GROUND
-5V			6	5			-5V
D7	1	55	8	7	55	1	D3/A19
D6	1	55	10	9	55	1	D2/A18
D5	1	55	12	11	55	1	D1/A17
D4	1	55	14	13	55	1	D0/A16
A15		55	16	15	55		A7
A14		55	18	17	55		A6
A13		55	20	19	55		A5
A12		55	22	21	55		A4
A11		55	24	23	55		A3
A10		55	26	25	55		A2
A9		55	28	27	55		A1
A8		55	30	29	55		A0
RD* (1)		55	32	31	55		WR* (1)
MEMRQ* (1)		55	34	33	55		IORQ* (1)
MEMEX			36	35	(4)		IOEXP
MCSYNC* (1)		55	38	37			REFRESH*
STATUS 0* (1)		55	40	39	55		STATUS 1* (1)
BUSRQ* (1)	1	55	42	41	55	1	BUSAK* (1)
INTRQ* (1)	1		44	43	55		INTAK* (1)
NMIRQ* (1)	1		46	45		1 (2)	WAITRQ*
PBRESET*	(3)		48	47	220	3	SYSRESET*
CNTRL*	1		50	49	55		CLOCK* (1)
PCI	3		52	51	18		PCO
AUX GND			54	53			AUX GND
AUX -V	IN		56	55		IN	AUX +V

* Low-level active logic
One LSTTL Load = -0.4 mA

(1) Drive and loading shown are with resistor pack R2 removed. R2 is installed on one multimaster per system; as shipped, add 10 TTL loads to input load and subtract 10 TTL loads from output drive.

(2) Add 21 TTL loads if R2 is installed (see note 1).

(3) Switch input

(4) Can be jumpered to ground.

Figure 1-8. STD Edge Connector

CLOCK SPECIFICATION

* Nominal Frequency: 4.9152 MHz
* Frequency Stability: ±.01% (includes initial tolerance and aging)

RS-422-A SERIAL INTERFACE SWITCHING CHARACTERISTICS

All typical values are at T=25°C and Vcc=5V

Line Driver (SN75174):

Parameter	Test Conditions	MIN	TYP	MAX	UNITS
Vo=VOH	VIH=2V; IOH=-33 mA		3.7		V
Vo*=VOL	VIH=2V; IOL=33 mA		1.1		V
Vo=VOL	VIL=0.8V; IOL=33 mA		1.1		V
Vo*=VOH	VIL=0.8V; IOH=-33 mA		3.7		V
Voc (common-mode)	RL=100 ohms		2.4	3.0	V
Vod (differential-mode)	RL=100 ohms	2.0	2.6		V
IOS (High)	Vo=Vcc			180	mA
IOS (Low)	Vo=-7V			-180	mA

Line Receiver (SN75175):

PARAMETER	MIN	TYP	MAX	UNITS
Vic (common mode)	-12		12	V
Vid (differential-mode)	-12		12	V
rin (input resistance)	12K			ohm

RS-422-A MAXIMUM BAUD RATE

* Synchronous: 1.229M baud
* Self-clocked: 307.2K baud
* Asynchronous: 76.8K baud

PIN NUMBER	SIGNAL FLOW		PIN NUMBER	SIGNAL FLOW	
SIGNAL					SIGNAL
TXC*=	OUT	2	1	OUT	TXC*+
TXD=	OUT	4	3	OUT	TXD+
GND	OUT	6	5	OUT	GND
RXD+	IN	8	7	IN	RXD=
RXC*+	IN	10	9	IN	RXC*=

Figure 1-9. RS-422-A Serial Connector J2

RS-232-C SERIAL INTERFACE SWITCHING CHARACTERISTICS

All typical values are at 25°C

Line Driver (MC1488): (AUX V+ = 12V; AUX V- = -12V)

Parameter	TEST CONDITIONS	MIN	TYP	MAX	UNITS
VOH	VIL=0.8V; RL=3K ohms	8.2	9.5		V
VOL	VIH=1.9V; RL=3K ohms	-8.1	-9.4		V
IOS (High)	Vi=0.8V; Vo=0	-6	-10	-12	mA
IOS (Low)	Vi=1.9V; Vo=0	6	10	12	mA
ro	AUX V+=0; AUX V-=0; -2V ≤ Vo ≤ 2V	300			ohms

Line Receiver (MC1489): (Vcc = 5V)

PARAMETER	MIN	TYP	MAX	UNITS
VIH	3.0			V
VIL			-3.0	V
rin		3.5K		ohms

*RS-232-C MAXIMUM BAUD RATE**

* Synchronous: 19.2K baud
* Asynchronous: 19.2K baud

***NOTE:** Baud rates may optionally be set at 38.4K or 76.8K. RS-232-C specification defines the maximum data rate as 19.2K baud.

PIN NUMBER			PIN NUMBER		
SIGNAL FLOW MNEMONIC				SIGNAL FLOW MNEMONIC	
		2	1		
RCLK (DB)	OUT	4	3	IN/OUT	TXD*(BA)
		6	5	IN/OUT	RXD*(BB)
RCLK (DD)	IN/OUT	8	7	IN/OUT	RTS (CA)
		10	9	IN/OUT	CTS (CB)
		12	11	IN/OUT	DSR (CC)
DTR (CD)	IN/OUT	14	13	OUT	SIGNAL GND (AB)
		16	15		
		18	17		
		20	19		
TCLK (DA)	IN/OUT	22	21		
		24	23		
		26	25		

Figure 1-10. RS-232-C Serial Connector J1

Mechanical Specifications

The 7863 meets all general mechanical specifications as given in the general *STD Series 7000 and 70C00 Specification*, Pro-Log document #111684. This document is included in the *Technical Manual and Data Book*, document #107855.

Card Spacing: requires one .5-inch center card slot.

SERIAL CONNECTORS

The following cables and connectors are not supplied with the 7863 but can be used for connecting external devices to the onboard serial communication channels:

Connector J1 (RS-232-C):

Pro-Log RC701-3 (3' flat cable terminated with 25-pin female D connector with RS-232-C pinout)

Pro-Log RC702-3 (3' flat cable terminated with 25-pin male D connector with RS-232-C pinout)

Connector J2 (RS-422-A):

FEMALE TRANSITION CONNECTORS	
Manufacturer	Part Number
T&B/Ansley	609-1030-1
Berg	65847-004-A
Burndy	FRS 10 DB-1P
Robinson-Nugent	1DS-10-SR
SAE	RD6310BRSC
3M	3473-7010

Timing Specifications

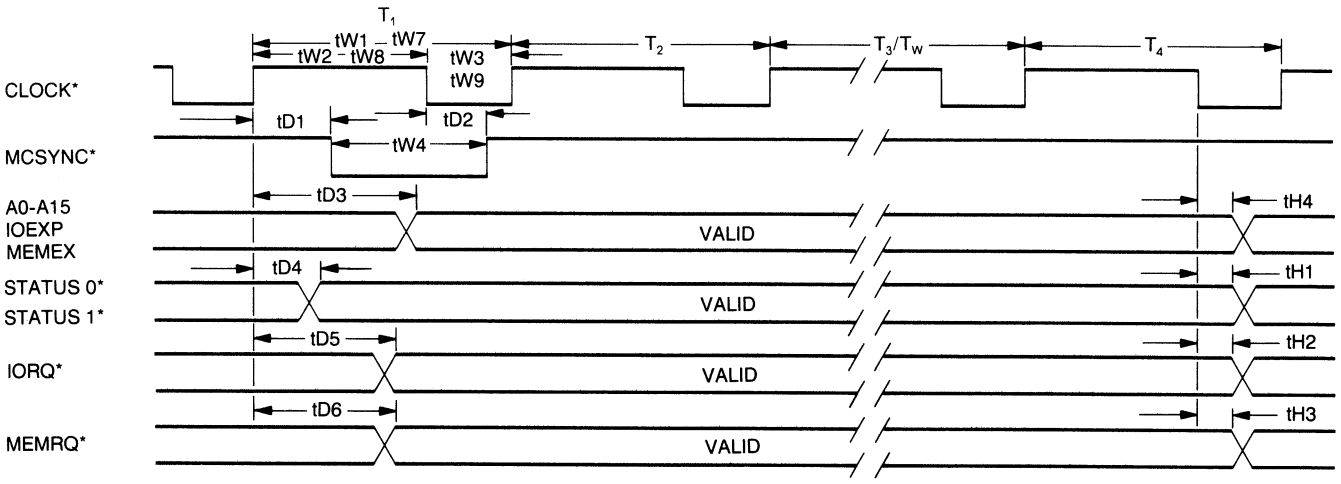
Timing for 7863 STD BUS interface signals is shown in Figures 1-11 through 1-17. The timing has been divided into seven functional groups: Bus Status, Bus Read, Bus Write, Bus Interrupt, Bus Wait, Bus Exchange Timing, and Multimaster. All times shown have been derated to include a temperature range of 0 to 65°C, a supply range of 4.75V to 5.25V, and a 300 pF load.

The critical timing parameters as defined by the proposed IEEE P961/D4 STD BUS standard (February 4, 1985 document, Figures 9 and 10) are listed below. The 7863 meets the September 15, 1985 STDMG STD-8088 revision 2.0 specification.

- tAR, maximum available Read Access Time, equals 210 nanoseconds.

- tSWD, minimum available Write Data Setup Time, equals 333 nanoseconds.

- tHWD, minimum available Write Data Hold Time, equals 60 nanoseconds.

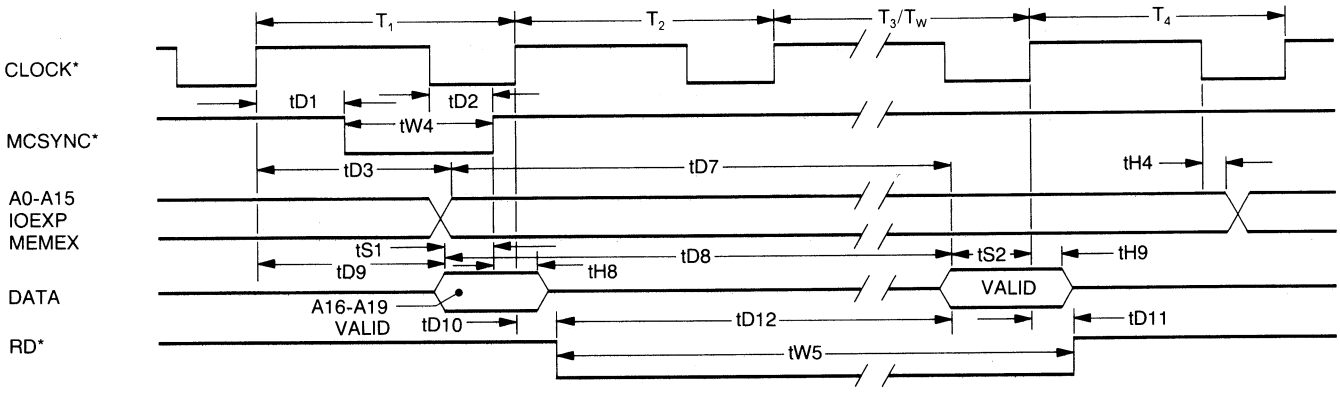


SYMBOL	PARAMETER	4.9152 MHz	
		MIN	MAX
tW1	8088 CLOCK* period	201	205
tW2	8088 CLOCK* high width	133	137
tW3	8088 CLOCK* low width	66	70

SYMBOL	PARAMETER	4.9152 MHz	
		MIN	MAX
tD1	Delay from CLOCK* to MCSYNC* low	121	
tD2	Delay from CLOCK* to MCSYNC* high		102
tD3	Delay from CLOCK* to Address 0-15, MEMEX, IOEXP		125
tD4	Delay from CLOCK* to STATUS		57
tD5	Delay from CLOCK* to IORQ*		57
tD6	Delay from CLOCK* to MEMRQ*		57
tHl	STATUS* hold after CLOCK*	12	
tH2	IORQ* hold after CLOCK*	12	
tH3	MEMRQ* hold after CLOCK*	4	
tH4	Address 0-15, MEMEX, IOEXP hold after CLOCK*	5	
tW4	MCSYNC* pulse width	68	

All times given in nanoseconds.

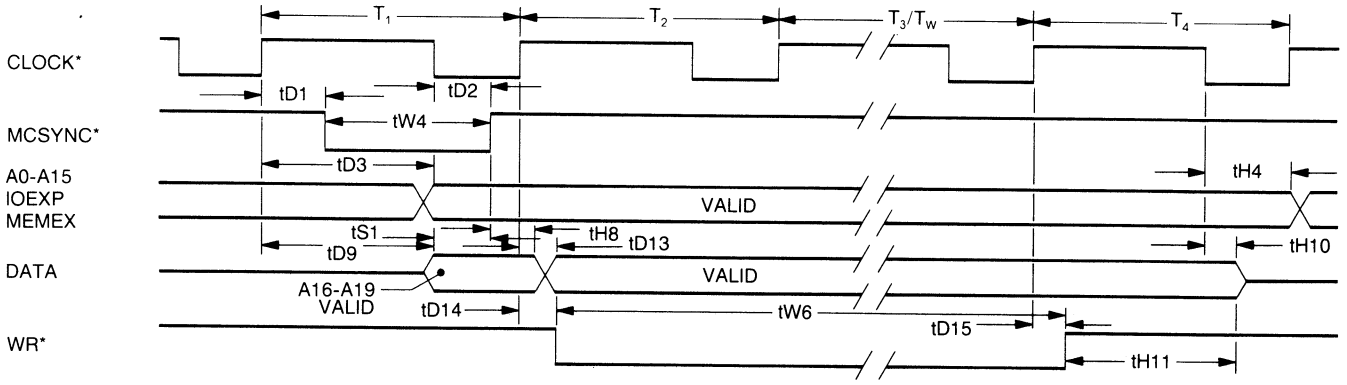
Figure 1-11. Bus Status Timing



SYMBOL	PARAMETER	4.9152 MHz	
		MIN	MAX
tD1	Delay from CLOCK* to MCSYNC* low	121	
tD2	Delay from CLOCK* to MCSYNC* high		102
tD3	Delay from CLOCK to Address 0-15		125
tD7	Delay from Address 0-15, MEMEX, IOEXP to data valid	433	
tD8	Delay from Address 16-19	433	
tD9	Delay from CLOCK* to Address 16-19		125
tD10	Delay from CLOCK* to RD* low	116	155
tD11	Delay from CLOCK* to RD* high	48	87
tD12	Delay from RD* to Data Valid	210	
tH4	Address 0-15, MEMEX, IOEXP hold after CLOCK*	5	
tH8	Address 16-19 hold after MCSYNC*	53	
tH9	Data hold after CLOCK*	6	
tS1	Address 16-19 setup to MCSYNC*	64	
tS2	Data setup to CLOCK*	52	
tW4	MCSYNC* pulse width	68	
tW5	RD* pulse width	339	

All times given in nanoseconds.

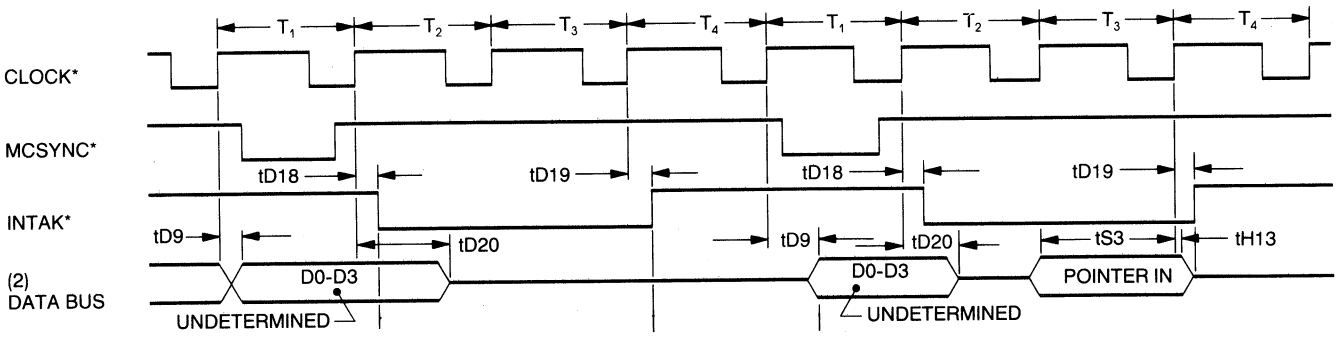
Figure 1-12. Bus Read Timing



SYMBOL	PARAMETER	4.9152 MHz	
		MIN	MAX
tD1	Delay from CLOCK* to MCSYNC* low	121	
tD2	Delay from CLOCK* to MCSYNC* high		102
tD3	Delay from CLOCK* to Address 0-15, MEMEX, IOEXP		125
tD9	Delay from CLOCK* to Address 16-19		125
tD13	Delay from CLOCK* to Data Valid		119
tD14	Delay from CLOCK* to Advanced WR* low	48	87
tD15	Delay from CLOCK* to Advanced WR* high	48	87
tH4	Address 0-I5, MEMEX, IOEXP hold after CLOCK*	5	
tH8	Address 16-19 hold after MCSYNC*	53	
tH10	Data hold after CLOCK*	10	
tH11	Data hold after Advanced WR* high	58	
tS1	Address 16-19 setup to MCSYNC*	64	
tW4	MCSYNC* pulse width	68	
tW6	Advanced WR* pulse width	407	

All times given in nanoseconds.

Figure 1-13. Bus Write Timing

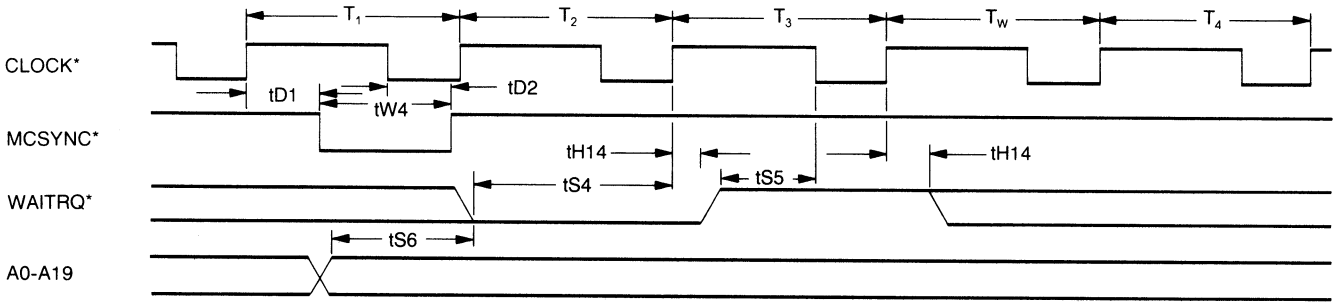


SYMBOL	PARAMETER	4.9152 MHz	
		MIN	MAX
tD9	Delay from CLOCK* to Address 16-19		125
tD18	Delay from CLOCK* to INTAK* low	116	155
tD19	Delay from CLOCK* to INTAK* high	48	87
tD20	Delay from CLOCK* to Data Bus 3-State		51
tH13	Interrupt vector hold from CLOCK*	6	
tS3	Interrupt vector setup to CLOCK*	52	

All times given in nanoseconds.

NOTE: The illustrated example does not show possible additional clock cycles between T4 of first cycle and T1 of second INTAK cycle. These clock cycles occur if the 8088 internal queue is not full. These clock cycles do not affect system operation, only total interrupt response time.

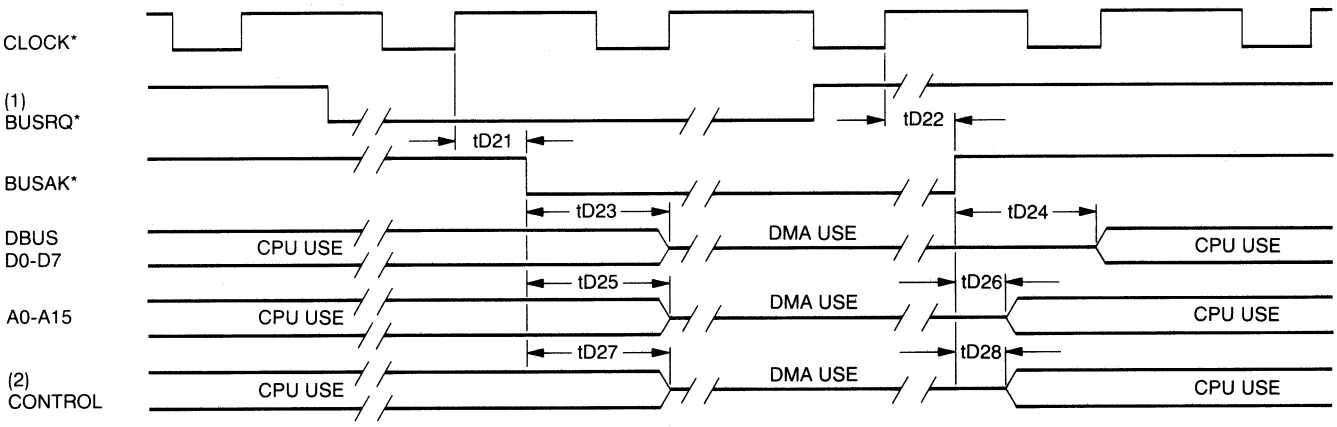
Figure 1-14. Bus Interrupt Timing



SYMBOL	PARAMETER	4.9152 MHz	
		MIN	MAX
tDl	Delay from CLOCK* to MCSYNC* low	121	
tD2	Delay from CLOCK* to MCSYNC* high		102
tH14	WAITRQ* hold after CLOCK*	9	
tS4	WAITRQ* low setup to CLOCK*	26	
tS5	WAITRQ* high setup to CLOCK*	26	
tS6	Address 0-19 setup to WAITRQ*	256	
tW4	MCSYNC* pulse width	68	

All times given in nanoseconds.

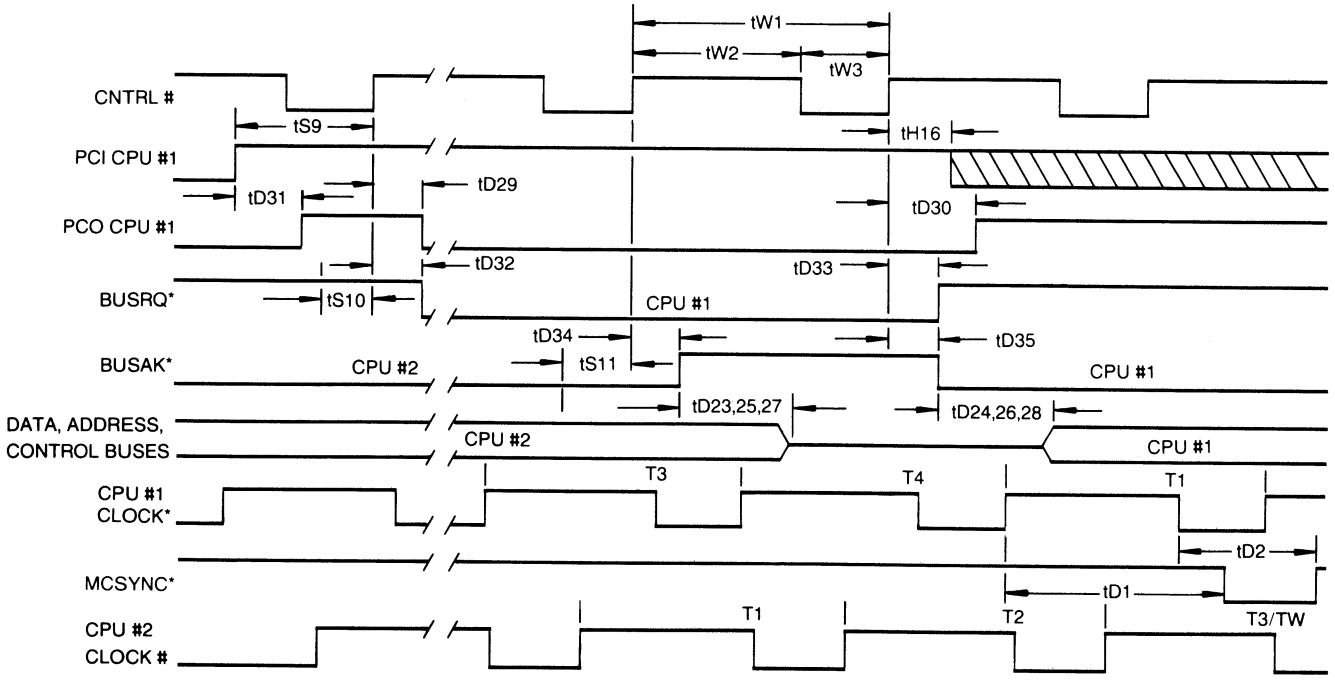
Figure 1-15. Bus Wait Request Timing



SYMBOL	PARAMETER	4.9152 MHz	
		MIN	MAX
tD21	Delay from CLOCK* to BUSAK* high	0	178
tD22	Delay from CLOCK* to BUSAK* low	0	178
tD23	Delay from BUSAK* to Data Bus 3-state	-63	
tD24	Delay from BUSAK* to Data Bus driven	288	
tD25	Delay from BUSAK* to Address Bus 3-state	-88	
tD26	Delay from BUSAK* to Address Bus driven	212	
tD27	Delay from BUSAK* to Control Bus 3-state	-49	
tD28	Delay from BUSAK* to Control Bus driven	283	

All times given in nanoseconds.
[1]Timings shown are for an asynchronous BUSRQ*.
[2]Control includes STD Bus lines MCSYNC*, IORQ, MEMRQ*, RD*, WR*, STATUS0*, STATUS1*.

Figure 1-16. Bus Exchange Timing



SYMBOL	PARAMETER	4.9152 MHz	
		MIN	MAX
tD1	Delay from CLOCK* to MCSYNC* low	0	
tD2	Delay from CLOCK* to MCSYNC* high	0	111
tD23	Delay from BUSAK* to Data Bus 3-State		40
tD24	Delay from BUSAK* to Data Bus Driven	0	
tD25	Delay from BUSAK* to Address Bus 3-State		40
tD26	Delay from BUSAK* to Address Bus Driven	0	
tD27	Delay from BUSAK* to Control Bus 3-State		40
tD28	Delay from BUSAK* to Control Bus Driven	0	
tD29	Delay from CNTRL* to PCO low		23
tD30	Delay from CNTRL* to PCO valid		23
tD31	Delay from PCI to PCO (per CPU)		10
tD32	Delay from CNTRL* to BUSRQ* low		65
tD33	Delay from CNTRL* to BUSRQ* high		40
tD34	Delay from CNTRL* to BUSAK* high		40
tD35	Delay from CNTRL* to BUSAK* low		40
tH16	PCI high hold after CNTRL*	0	
tS9	PCI high setup to CNTRL*	25	
tS10	BUSRQ* setup to CNTRL*	25	
tS11	BUSAK* setup to CNTRL*	25	
tW1	CNTRL* period	200	
tW2	CNTRL* high	113	
tW3	CNTRL* low	65	

All times given in nanoseconds
NOTE: 16 Multimasters supported.

Figure 1-17. Multimaster Timing

INTRODUCTION

This section describes the procedures for installing one or more 7863-1 cards in an STD BUS system and the jumper options that must be considered prior to installation. Since a multimaster STD system is somewhat more complex than a traditional single-CPU system, review this section carefully to ensure that each 7863-1 is properly configured before operation.

POWER SUPPLY

CAUTION

To prevent possible damage to your STD BUS system, make sure power is off before inserting a card into the card rack or before removing a card from the card rack.

CAUTION

To guarantee proper operation of the 7863 card, LOGIC GND and AUX GND must be connected at the power supply.

Pro-Log power supplies are shipped with both GNDs tied together and therefore meet this requirement. If a source other than Pro-Log is used for power supplies, care must be taken to ensure this requirement is met.

In selecting a power supply for an STD BUS system that uses one or more 7863 cards, make sure that all electrical requirements of the 7863 as specified in Section 1 (Electrical Specifications) are met. Each 7863 can draw more than 2.4 amps of current from the 5V supply (Vcc) with memory installed. The power supply selected, the power distribution cables and connectors, and the STD BUS backplane must be able to supply the required current to each card while maintaining Vcc within the ±5% tolerance specified.

As shipped, the 7863 is held in a reset state until 60 milliseconds after Vcc reaches 4.75V minimum during power-up, ensuring that the card is operated within specifications. Due to component tolerances in the reset circuitry, the actual level at which reset is released and operation begins may be as high as 4.90V. Voltage drops in cabling, connectors, and the STD BUS backplane may require that the power supply be adjusted before reset is released. See the following information on Reset, Jumper W5, for more information.

PHYSICAL INSTALLATION

The 7863 operates as part of an STD BUS system. A 7863 can be inserted into any slot in the STD BUS card rack, although multimaster systems require certain physical arrangements of cards in the card rack. These arrangements are discussed in detail under "Priority Chain." When installed, the card ejector tab must be at the top as shown in Figure 2-1. If access to the card is needed for testing or servicing, a card extender such as the Pro-Log 7911 can be used. Card extenders tend to degrade STD BUS signals, which may cause problems with dynamic RAM or other STD cards with stringent bus timing or noise requirements. The use of more than one or two extenders may cause problems in any system.

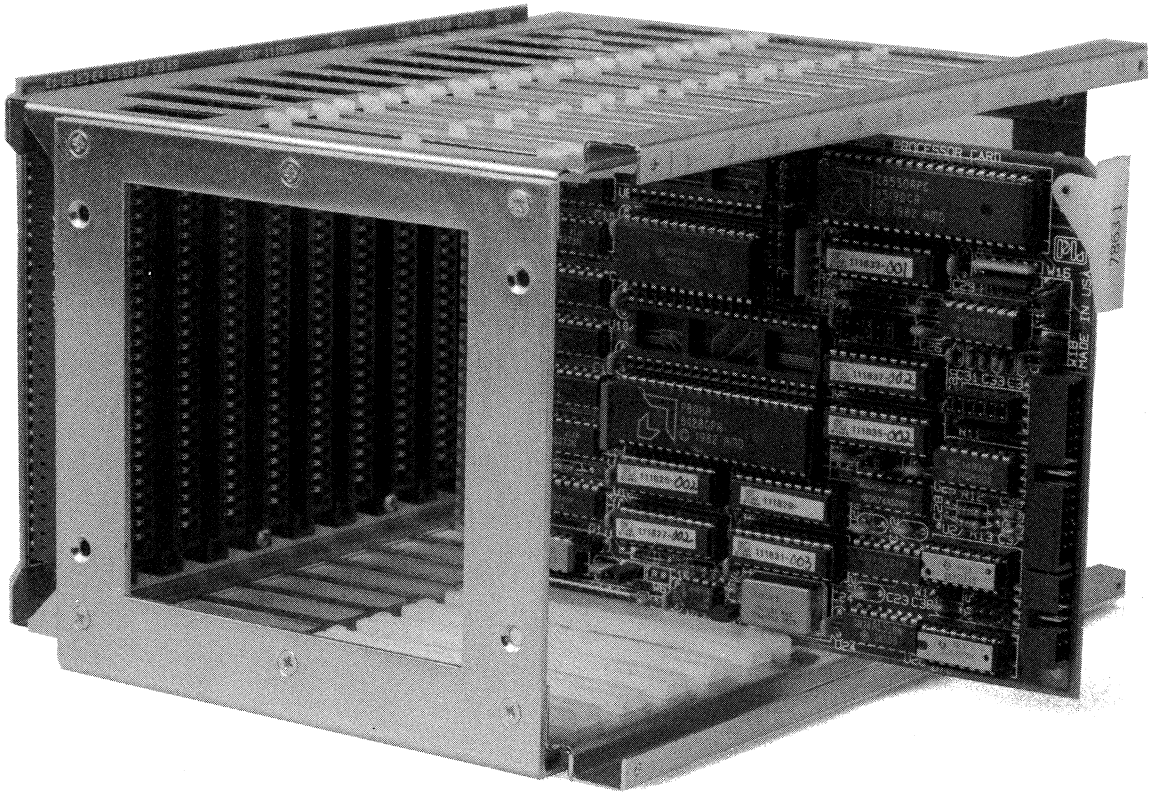


Figure 2-1. Installation

Priority Chain

The priority chain lines on the STD BUS (PCO, pin 51, and PCI, pin 52) are used for bus arbitration in a multimaster system. Multiple 7863 and/or other multimaster CPU cards must be installed in the card rack so the priority chain is continuous between them, i.e., PCO of one multimaster drives PCI of the next. Multimaster CPU cards are usually installed in adjacent slots in the card rack (see Figure 2-2A). If a non-multimaster card (such as a memory or I/O card) must be installed between two multimasters, it must not use the priority chain on-card but must pass the chain through by connecting PCI to PCO onboard (see Figure 2-2B).

The PCI input to the rightmost multimaster in the STD card rack should never be driven, since the rightmost multimaster has the highest priority of all multimasters in the system. Usually this is accomplished by installing the multimaster CPU cards starting with the rightmost slot in the card rack (see Figure 2-2A, B, and D). If this cannot be done, it is necessary to leave the card slot immediately to the right of the rightmost multimaster vacant (see Figure 2-2C) if any card to the right of the multimasters drives or uses the priority chain.

STD BUS cards that use the priority chain for another purpose (such as I/O cards that use the chain for interrupt prioritization) can be used in the same card rack with 7863 or other multimaster CPU cards. To do this, the priority chain must be broken, either by leaving a vacant card slot between the multimasters and the other cards (see Figure 2-2C) or by removing the shorting connector from a jumper (W4 on the 7863) so the PCO output of the lowest-priority (leftmost) multimaster is not driven (see Figure 2-2D). The system then has two separate priority chains with cards using each in separate sections of the card rack.

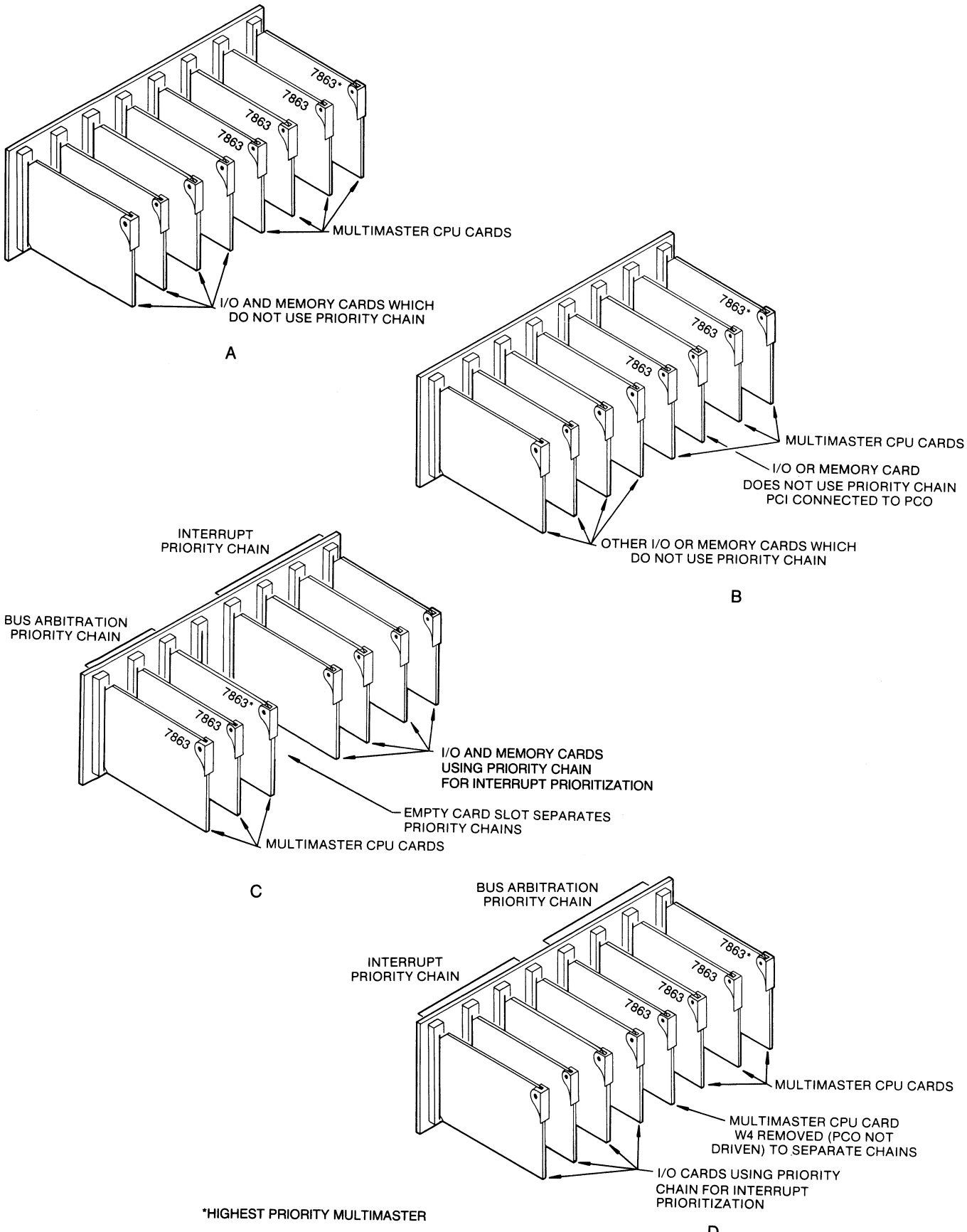


Figure 2-2. Multimaster Installation Options

CARD LAYOUT AND JUMPERS

The 7863 has onboard jumpers to configure the card to meet system requirements. The jumper configurations consist of wire jumpers and sockets, as shown in Figure 2-3. The wire jumpers can be removed and installed without the use of soldering equipment. Figure 2-4 lists sources for the shorting connectors.

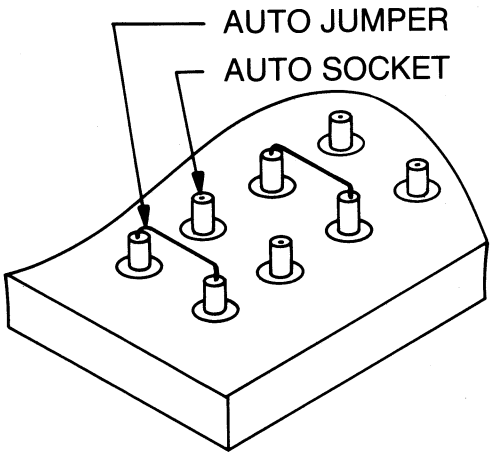


Figure 2-3. Jumper Configuration

SOURCE	PART NUMBER
Fancort Industries, Inc.	J 0.100X0.250 B 24
Squires Electronics, Inc.	J 0.100X0.250 B 24

Figure 2-4. Sources for Shorting Jumpers

U16 PAL Assembly Installation

PAL assembly 111829-xxx is installed in socket U16 for the 7863-1 multimaster card only.

Physical Layout and Jumper Options

Figure 2-5 on the following pages provides a reference for the location and a brief description of each jumper on the 7863. The discussion following Figure 2-5 divides the jumper options into six functional groups: Multimaster Configuration, Reset, Memory and I/O, Interrupt, RS-422-A Serial Port, and RS-232-C Serial Port.

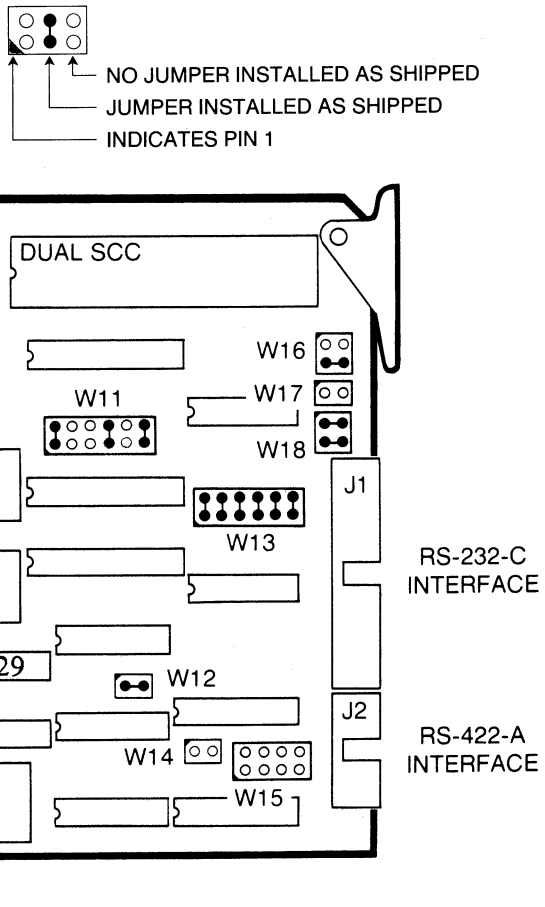


Figure 2-5. Physical Layout and Jumper Options

JUMPER GROUP	JUMPER INSTALLED	JUMPER REMOVED	DESCRIPTION	AS SHIPPED	SEE PAGE
W1	1-2 —	— 1-2	IOEXP DRIVE ENABLE • IOEXP grounded • IOEXP driven by another card	*	2-9
W2	1-2 —	— 1-2	CNTRL* SOURCE SELECT • Driven by on-card clock generator • Driven by external source	*	2-7
W3	1-2 —	— 1-2	BUS INTRQ* ENABLE • Enable • Disable	*	2-10
W4	1-2 —	— 1-2	PCO DRIVE ENABLE • PCO driven • PCO not driven	*	2-7
W5	1-2 —	— 1-2	RESET VOLTAGE SELECT • Power-on reset level 4.75V min. • Power-on reset level 4.75V max.	*	2-8
W6	—	—	PRO-LOG USE ONLY	—	—
W8	1-2, 9-10 1-2, 9-10 1-2, 3-4, 9-10 3-4, 5-6, 9-10 1-2, 7-8 2-4, 7-8, 9-10	3-4, 5-6, 7-8 3-4, 5-6, 7-8 5-6, 7-8 1-2, 7-8 3-4, 5-6, 9-10 1-3, 5-6	MEM 1 CONFIGURATION • 8K ROM • 16K ROM • 32K ROM • 64K ROM • 8K RAM • 32K RAM	*	2-8
W9	1-2, 9-10 1-2, 9-10 1-2, 3-4, 9-10 3-4, 5-6, 9-10 1-2, 7-8 2-4, 7-8, 9-10	3-4, 5-6, 7-8 3-4, 5-6, 7-8 5-6, 7-8 1-2, 7-8 3-4, 5-6, 9-10 1-3, 5-6	MEM 0 CONFIGURATION • 8K ROM • 16K ROM • 32K ROM • 64K ROM • 8K RAM • 32K RAM	*	2-8
W10	1-2 1-3 2-4 —	3-4 2-4 1-3 1-2, 3-4	LOCK MODE SELECT • DMA lock enable • Hardware lock enable • Lock always asserted (single CPU system) • Software Lock only	*	2-7
W11	1-2, 3-4, 5-6 1-2, 3-4 1-2, 5-6 1-2 3-4, 5-6 3-4 5-6 —	— 5-6 3-4 3-4, 5-6 1-2 1-2, 5-6, 11-12 1-2, 3-4, 9-10, 11-12 1-2, 3-4, 5-6	MEMORY MAPPING MEM 0: (W11, Pins 1-2, 3-4, 5-6) • 00000-0FFFF (64K) • 00000-07FFF (32K) • 00000-03FFF (16K) • 00000-01FFF (8K) • E0000-EFFFF (64K) • F0000-F7FFF (32K) • FC000-FDFFF (8K) • SOCKET DISABLED	*	2-9
	7-8, 9-10, 11-12 7-8, 9-10 7-8, 11-12 7-8 9-10, 11-12 9-10 11-12 —	— 11-12 9-10 9-10, 11-12 7-8 5-6, 7-8, 11-12 3-4, 5-6, 7-8, 9-10 7-8, 9-10, 11-12	MEM 1: (W11, Pins 7-8, 9-10, 11-12) • F0000-FFFFF (64K) • F8000-FFFFF (32K) • FC000-FFFFF (16K) • FE000-FFFFF (8K) • 10000-1FFFF (64K) • 08000-0FFFF (32K) • 02000-03FFF (8K) • SOCKET DISABLED	*	2-9
W12	1-2 —	— 1-2	INTRQ* VECTOR SOURCE SELECT • Bus vector during INTAK* • SCC provides vector during INTAK*	*	2-10
W13	1-2, 3-4, 5-6 7-8, 9-10, 11-12 1-3, 2-4, 5-7 6-8, 9-11, 10-12	— — 	RS-232-C CONFIGURATION • DCE • DTE	*	2-12
W14	— 1-2	1-2 —	RS-422-A TCLK/BITBUS RTS SELECT • Transmit Clock enable • Bit Bus RTS enable	*	2-12
W15	— 1-2, 3-4	1-2, 3-4 —	RS-422-A CONFIGURATION—DATA • Separate Receive & Transmit data lines • Receive & Transmit data lines connected for multidrop	*	2-11
	— 5-6, 7-8	5-6, 7-8 —	RS-422-A CONFIGURATION—CLOCK • Separate Receive & Transmit clock lines • Transmit clock connected to receive clock for multidrop	*	2-11
W16	3-4 1-2	1-2 3-4	RS-422-A RECEIVE CLOCK SOURCE • External (from RS-422-A interface) • SCC PCLK	*	2-11
W17	1-2	—	RS-232-C DTE CLOCK • DTE transmit clock driven by 7863 (DCE) • DTE transmit clock not driven	*	2-13
W18	1-2, 3—4 —	— 1-3, 2-4	RS-232-C CONFIGURATION • DCE • DTE	*	2-12
W19-20	1-2	2-3	PRO-LOG USE ONLY	—	—

Figure 2-5. (Continued)

Multimaster Configuration

Jumpers W2, W4, and W10 and Resistor Pack R2 affect the operation of the 7863 with other multimaster CPU cards in an STD BUS system. They must be configured correctly to ensure proper operation of the bus arbitration logic. Figure 2-6 shows how these jumpers should be configured on each 7863 card in a multimaster system or on a 7863 used in a single-CPU system. Figures 2-7 through 2-9 and their respective discussions describe these jumpers individually.

RESISTOR PACK R2, CONTROL PULL-UPS (Figure 2-6)

Resistor pack R2 contains 1000-ohm pull-up resistors which are required on certain STD BUS control signals (INTRQ*, NMIRQ*, WAITRQ*, INTAK*, RD*, WR*, MEMRQ*, CLK*, MCSYNC*, BUSRQ*, and BUSAK*) in a multimaster system. This resistor pack should be installed on one multimaster card in a card rack. The resistor pack containing these pull-up resistors should be removed from all other multimasters to prevent excessive loading of these control signals.

7863 #	DESCRIPTION	JUMPER W2	JUMPER W4	JUMPER W10	RESISTOR PACK R2
1	Multiple CPU System: Highest priority (Rightmost in rack)	1-2*	1-2	1-2 DMA Lock Mode or 1-3 Hardware Lock Mode or None Software Lock only	INSTALLED
2 ↕ n-1		None	1-2	1-3 Hardware Lock Mode or None Software Lock only	REMOVED
n (2-16)	Lowest priority (Leftmost in rack)	None	None	1-3 Hardware Lock Mode or None Software Lock only	REMOVED
1	Single CPU System	1-2	None	2-4 DMA Lock Always asserted	INSTALLED

*Jumper pins with shorting connectors installed.

Figure 2-6. Multimaster Configuration Jumpers

CNTRL SOURCE SELECT, JUMPER W2 (Figure 2-7)*

This jumper selects whether the clock (CNTRL*, Pin 50 on the STD BUS) used by all multimaster CPU cards in the system for bus arbiter synchronization is provided by this card or by another multimaster. Only one multimaster in the system can drive CNTRL* and it should be the highest priority multimaster in the system (the rightmost multimaster installed in the card rack). If the 7863 is to be the highest priority multimaster or is to be used as the only CPU card in the system, a shorting connector should be installed in Jumper W2; if not, the shorting connector should be removed.

Description	Pins With Shorting Connectors Installed
This card drives CNTRL*	1-2**
Another multimaster drives CNTRL*	None

**As shipped

Figure 2-7. CNTRL* Source Select, Jumper W2

PCO DRIVE ENABLE, JUMPER W4 (Figure 2-8)

This jumper determines whether the priority chain output (PCO, Pin 51 on the STD BUS) is driven by the 7863 bus arbiter or left floating. When a shorting connector is installed in W4, PCO is driven. For proper bus arbitration, all 7863 cards in a multimaster

STD system must have this jumper installed unless the 7863 is the lowest priority multimaster (the leftmost multimaster installed in the card rack). The shorting connector must be removed from Jumper W4 on this card if peripheral or memory cards installed to the left of this lowest priority multimaster use the priority chain for some other purpose (such as interrupt prioritization). Removal of the shorting connector effectively splits the STD BUS priority chain into two separate chains that can be used for different purposes.

Description	Pins With Shorting Connectors Installed
PCO is driven by this card	1-2**
PCO is not driven	None

**As shipped

Figure 2-8. PCO Drive Enable, Jumper W4

LOCK MODE SELECT, JUMPER W10 (Figure 2-9)

The 7863 has an onboard one-bit output port (mapped as port FFFF, bit D0) that can be set to lock the bus, i.e., to prevent any other multimaster from taking control of the STD BUS away from the 7863 until the port has been cleared. Jumper W10 enables this bus lock function and determines whether Hardware Lock Mode or DMA Lock Mode is active when the output port bit has been set.

If no shorting connector is installed in Jumper W10, the lock control port is disabled. If a shorting connector is installed in W10 pins 1-2, Hardware Lock will be active when the output port bit is set. Any number of 7863s or other multimasters in a system can each implement this bus lock mode.

If a shorting connector is installed in W10 pins 1-3, DMA Lock Mode is active when the output port bit is set. This lock mode can only be used by the highest priority multimaster in the system (the rightmost multimaster installed in the card rack). When DMA Lock Mode is selected, no other multimaster can gain access to the bus, as with Hardware Lock Mode. In addition, BUSRQ* and BUSAK* on the STD BUS are no longer used for bus arbitration but are available for use by DMA controllers in normal BUSRQ*/BUSAK* operation. Bus access requests from all other multimasters are inhibited by disabling the STD BUS priority chain so only the highest priority multimaster can implement this mode. DMA operations in a multimaster STD system are only possible when the DMA Lock Mode is active.

In a single CPU system, it may be desirable to permanently select the DMA Lock Mode since bus arbitration will never be required, and normal DMA operations will be supported; DMA Lock Mode is always active when a shorting connector is inserted in Jumper W10, pins 2-4.

In addition to these bus lock modes, a software lock mode is always available. In this mode, a 7863 Multimaster CPU can retain control of the bus for the duration of a multicycle instruction, through the use of a lock prefix to the instruction in the program being executed.

Description	Pins With Shorting Connectors Installed
DMA Lock is enabled	1-2**
Hardware Lock is enabled	1-3
DMA Lock is always asserted	2-4
Hardware and DMA Lock disabled; software lock only	None

**As shipped

Figure 2-9. Lock Mode Select, Jumper W10

RESET, JUMPER W5 (Figure 2-10)

This jumper sets the voltage level to which Vcc on-card must rise before reset (and SYSRESET* on the STD BUS) is released, after a time delay. As shipped, with a shorting connector installed in Jumper W5 pins 1-2, Vcc must reach +4.75V minimum. This ensures that all devices on the 7863 are operating within specification (Vcc = 5V ±5%) when reset is released. Due to component tolerances, the actual voltage level at which reset is released may be as high as 4.90V. Because of voltage drops in cabling and in the STD BUS backplane and card edge connectors, Vcc on the card may never rise above 4.90V and the card will not come out of the reset state. Adjustment of the power supply or remote sensing of Vcc on the backplane may be required.

Removal of the shorting connector from Jumper W5 lowers the Vcc voltage level at which reset is released to 4.75V maximum, 4.60V minimum. Adjustment of the power supply may not then be necessary. Note that if this shorting connector is removed, reset may be released before Vcc has risen above 4.75V and the card may be operated out of specification. Pro-Log recommends that you leave the W5 shorting connector in place and adjust Vcc at the power supply if necessary to ensure correct operation.

Description	Pins With Shorting Connectors Installed
Power-on Reset level is 4.75V min.	1-2**
Power-on Reset level is 4.75V max.	None

**As shipped

Figure 2-10. Reset Voltage Select, Jumper W5

Memory and I/O

The 7863 has two memory sockets: MEM1 and MEM0. Jumpers W8 and W9, respectively, determine what type and size of memory device can be installed in each socket and Jumper W11 determines where each device is mapped in the 8088 CPU one-megabyte address range.

The 7863 memory sockets can accept JEDEC-standard byte-wide ROMs and RAMs with an access time of 300 nanoseconds or less. Refer to Figure 2-11 for a list of recommended memory devices.

The only jumper that affects I/O is W1, which selects whether or not IOEXP, pin 35 on the STD BUS, is grounded onboard the 7863.

MEMORY TYPE	MEMORY SIZE	MANUFACTURER'S PART NUMBER	ACCESS TIME (nS)
EPROM	8K	Intel 2764	300 max
	16K	Intel 27128 AMD 27128	
	32K	Intel 27256 AMD 27256	
	64K	Intel 27512 AMD 27512	
RAM	8K	Hitachi HM6264 LP-15 Toshiba TC5565 PL-15	200 max
	32K	NEC D43256-15	

(1) This device not available at publication

Figure 2-11. Recommended Memory Devices

MEMORY SOCKET CONFIGURATION, JUMPERS W8 AND W9 (Figure 2-12)

These jumpers select what type of device is to be installed in sockets MEM1 and MEM0. Each socket can be independently configured to accept 8K, 16K, 32K, or 64K ROMs; or 8K or 32K RAMs. Jumper W8 configures socket MEM1 and W9 configures socket MEM0.

DESCRIPTION	PINS WITH SHORTING CONNECTORS INSTALLED
8K ROM	1-2
16K ROM	1-2, 9-10[1]
32K ROM	1-2, 3-4, 9-10
64K ROM	3-4, 5-6, 9-10
8K RAM	1-2, 7-8[2]
32K RAM	2-4, 7-8, 9-10

[1]As shipped, W13 (MEM 1)
[2]As shipped, W14 (MEM 0)

Figure 2-12. Memory Socket Configuration Jumpers W8 (MEM1) and W9 (MEM0)

MEMORY MAPPING, JUMPER W11 (Figure 2-13)

Jumper W11 determines the mapping of the memory devices installed in sockets MEM1 and MEM0. Shorting connectors installed in pins 1 through 6 determine the mapping of the device in MEM0; those installed in pins 7 through 12 determine the mapping of the device in MEM1. There are eight mapping options for each socket. If the socket is not to be used (no memory device is to be installed), the last option should be selected disabling the socket and removing it from the memory map.

To prevent the selection of two options that map both sockets into the same address range, certain combinations of options for the two sockets cannot be used. For example, if MEM0 Option 5 is selected, then MEM1 Options 0, 2, 4, and 6 cannot be used. (MEM0 Option 5 requires that no shorting connector be installed in W11 pins 11-12, while MEM1 Options 0, 2, 4, and 6 require the installation of this jumper.) Unusable combinations occur only when Option 5 or 6 is selected for either socket; these options require that certain shorting connectors not be installed.

For the most efficient use of memory space, the selected mapping option should use a block of memory of the same size as the device to be installed in the socket. For example, Option 2 for socket MEM0 maps the socket at the address block 00000-03FFF hex, a 16K block of memory; a 16K memory device should be used. If an 8K device is installed, it will be redundantly addressed in the upper half of the 16K block. If a 32K device is installed, the upper 16K of the device cannot be accessed. Additionally, two options should never be selected that map both sockets into the same address block.

DESCRIPTION			PINS WITH SHORTING CONNECTORS INSTALLED	
SOCKET	OPTION	ADDRESS RANGE	PINS 1-6	PINS 7-12
MEM 0:	0	00000-0FFFF (64K BLOCK)	1-2,3-4,5-6	DON'T CARE
	1	00000-07FFF (32K BLOCK)	1-2,3-4	
	2	00000-03FFF (16K BLOCK)	1-2,5-6	
	3	00000-01FFF (8K BLOCK)**	1-2	
	4	E0000-EFFFF (64K BLOCK)	3-4,5-6	
	5	F0000-F7FFF (32K BLOCK)	3-4	(No connector in 11-12)
	6	FC000-FDFFF (8K BLOCK)	5-6	(No connector in 9-10, 11-12)
	7	SOCKET DISABLED	NONE	DON'T CARE
MEM 1:	0	F0000-FFFFF (64K BLOCK)	DON'T CARE	7-8,9-10,11-12
	1	F8000-FFFFF (32K BLOCK)		7-8,9-10
	2	FC000-FFFFF (16K BLOCK)**		7-8,11-12
	3	FE000-FFFFF (8K BLOCK)		7-8
	4	10000-1FFFF (64K BLOCK)		9-10,11-12
	5	08000-0FFFF (32K BLOCK)	(No connector in 5-6)	9-10
	6	02000-03FFF (8K BLOCK)	(No connector in 3-4, 5-6)	11-12
	7	SOCKET DISABLED	DON'T CARE	NONE

**As shipped

Figure 2-13. Onboard Memory Mapping, Jumper W11

IOEXP DRIVE, JUMPER W1 (Figure 2-14)

If a shorting connector is installed in Jumper W1 pins 1-2, IOEXP (pin 35 on the STD BUS) is connected to ground. If the connector is removed, IOEXP is not connected on the 7863 and can be driven by some other card in the system if necessary. IOEXP is defined in the STD BUS Specification as an enable line (low-level active) for I/O cards that decode an 8-bit I/O address instead of the 8088's full 16-bit address.

DESCRIPTION	PINS WITH SHORTING CONNECTORS INSTALLED
IOEXP grounded	1-2*
IOEXP driven by another card	NONE

**As shipped

Figure 2-14. IOEXP Drive Enable, Jumper W1

Interrupt Options

Jumpers W3 and W12 determine the response of the 7863 to maskable interrupt requests from the STD BUS (INTRQ*, pin 44).

INTRQ ENABLE, JUMPER W3 (Figure 2-15)*

If a shorting connector is installed in Jumper W3 pins 1-2, the 7863 can respond to INTRQ* on the STD BUS. If the connector is removed, the 7863 does not respond to INTRQ*. This jumper is used to select which multimasters in a system respond to STD BUS INTRQ*.

DESCRIPTION	PINS WITH SHORTING CONNECTORS INSTALLED
This card responds to STD BUS INTRQ*	1-2**
Does not respond to STD BUS INTRQ*	NONE

**As shipped

Figure 2-15. INTRQ* Enable, Jumper W3

INTRQ VECTOR SOURCE SELECT, JUMPER W12 (Figure 2-16)*

The 8088 CPU responds to a maskable interrupt request by executing an interrupt acknowledge cycle during which an 8-bit vector is read in on the processor data bus. This vector is normally provided by the in-

terrupting device and is used by the processor to locate the correct interrupt service routine. If a shorting connector is installed in Jumper W12 pins 1-2, the 8088 executes an interrupt acknowledge cycle and reads a vector (which must be supplied by the interrupting device) from the STD data bus in response to an STD BUS INTRQ*. Since bus access is required to read this vector and only one multimaster can have access to the bus during the interrupt acknowledge cycle when the interrupting device places its vector on the bus, only one multimaster in the system can have this jumper installed.

When the shorting connector is removed from Jumper W12, INTRQ* can no longer directly cause an 8088 maskable interrupt. However, INTRQ* is permanently connected (through a buffer) to an unused handshake line input of the Serial Communications Controller (SCC). The SCC can be programmed to generate an onboard interrupt in response to transitions on that handshake line and it provides a vector on-card when the interrupt is acknowledged. Since the source of the interrupt is on-card, access to the STD BUS is not required during the interrupt acknowledge cycle, so any or all multimasters in the system can simultaneously respond to STD BUS INTRQ* in this way.

DESCRIPTION	PINS WITH SHORTING CONNECTORS INSTALLED
Vector read from STD BUS during INTAK*	1-2**
SCC provides vector during INTAK*	NONE

**As shipped

Figure 2-16. INTRQ* Vector Source Select, Jumper W12

RS-422-A Serial Communications Channel

Jumpers W14, W15, and W16 affect the RS-422-A interface, which is connected to channel B of the SCC. The pinout of the RS-422-A connector eliminates the need for DCE/DTE configuration jumpers and preserves the differential signals on twisted pairs. Figure 2-17 shows how installation of the RS-422-A cable configures the channel as either DCE or DTE. For further details, see Pro-Log Application Note #140.

Termination for the RS-422-A receive data and receive clock signals is provided by 100-ohm resistors R12 and R13. If a different resistance value is required to match line impedance, these resistors should be replaced. In multidrop applications, termination is necessary only at the two ends of the signal line. All other termination resistors should be removed since they will appear in parallel and reduce the signal.

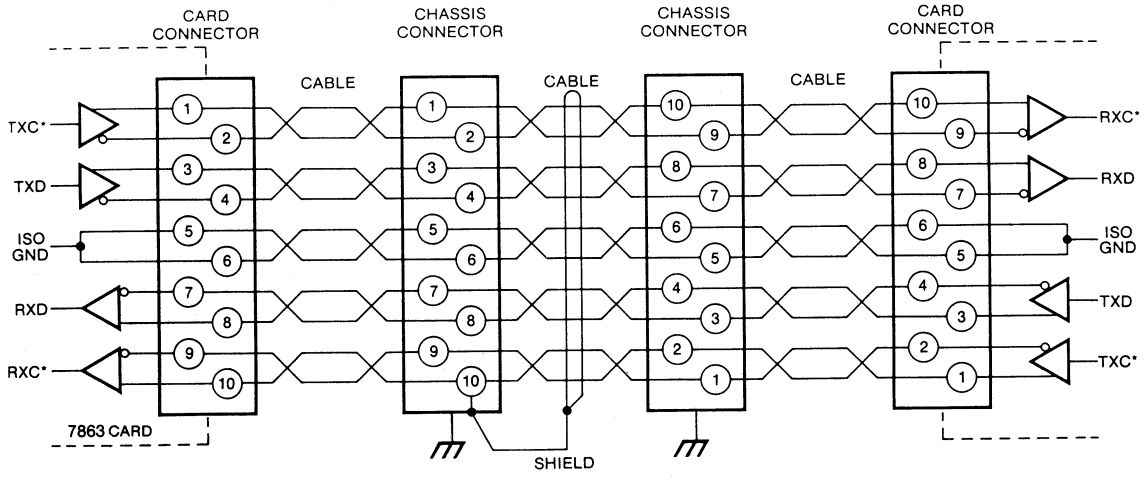


Figure 2-17. RS-422-A Synchronous System Interconnection

MULTIDROP CONFIGURATION, JUMPER W15 (Figure 2-18)

The 7863 RS-422-A interface meets RS-485 standards which allow up to 32 driver/receiver pairs to be placed on a single differential line. To configure the RS-422-A port for multidrop, shorting connectors should be installed in Jumper W15, pins 1-2 and 3-4, to connect the receive and transmit data lines. For synchronous operation, shorting connectors should also be installed in pins 5-6 and 7-8 to connect the receive and transmit clock lines. For applications other than multidrop, all shorting connectors should be removed.

DESCRIPTION	PINS WITH SHORTING CONNECTORS INSTALLED
Separate data and clock signals	None**
Asynchronous multidrop configuration: RxD and TxD connected	1-2,3-4
Synchronous multidrop configuration: RxD and TxD connected RxC and TxC connected	1-2, 3-4, 5-6, 7-8

Figure 2-18. RS-422-A Configuration, Jumper W15

RS-422-A RECEIVE CLOCK SOURCE, JUMPER W16 (Figure 2-19)

Normally the source for the RS-422-A channel receive clock is the signal RCLK from the RS-422-A connector. This option is selected by installing a shorting connector in Jumper W16, pins 3-4. In self-clocking synchronous modes, it is possible to increase the maximum possible data rate by a factor of four (to 307.2K baud) by bypassing the SCC internal baud rate generator and using the SCC clock input (PCLK, pin 20 on the SCC) as the receive clock source. (The receive clock is an input to the internal phase-locked loop in self-clocking modes.) Moving the shorting connector installed in Jumper W16 from pins 3-4 to pins 1-2 connects PCLK to the receive clock input for this purpose.

DESCRIPTION	PINS WITH SHORTING CONNECTORS INSTALLED
RCLK driven from RS-422-A interface	3-4**
RCLK driven by SCC PCLK signal	1-2

**As shipped

Figure 2-19. RS-422-A Receive Clock Source, Jumper W16

BITBUS™ TRANSCEIVER CONTROL ENABLE, JUMPER W14 (Figure 2-20)

When a shorting connector is inserted in Jumper W14 pins 1-2, the RS-422-A transmit clock lines TCLK+ and TCLK- can be used as the BITBUS Request To Send signal (RTS). This signal is used for transceiver control in a self-clocked BITBUS interconnect with repeaters. The BITBUS Interconnect Specification is available in the Distributed Control Modules Databook from Intel Corporation.

DESCRIPTION	PINS WITH SHORTING CONNECTORS INSTALLED
Normal RS-422-A Transmit Clock operation	None**
RS-422-A TCLK lines used for BITBUS transceiver control (RTS signal)	1-2

**As shipped

Figure 2-20. RS-422-A TCLK/RTS Select, Jumper W14

RS-232-C Serial Communications Channel

The RS-232-C interface is connected to channel A of the Serial Communications Controller.

RS-232-C CONFIGURATION, JUMPERS W13 AND W18 (Figure 2-21)

These jumpers select whether the channel operates as data communications equipment (DCE) or data terminal equipment (DTE). The orientation of the shorting connectors installed in Jumpers W13 and W18 determine which mode of operation is selected (as shown in Figure 2-22). As shipped, the channel is configured as DCE, since it will be connected to a terminal in many applications.

DESCRIPTION	PINS WITH SHORTING CONNECTORS INSTALLED	
	W13	W18
Configured as DCE	1-2, 3-4, 5-6 7-8, 9-10, 11-12**	1-2, 3-4**
Configured as DTE	1-3, 2-4, 5-7 6-8, 9-11, 10-12	1-3, 2-4

**As shipped

Figure 2-21. RS-232-C Configuration, Jumpers W13 and W18

™BITBUS is a registered trademark of Intel Corporation.

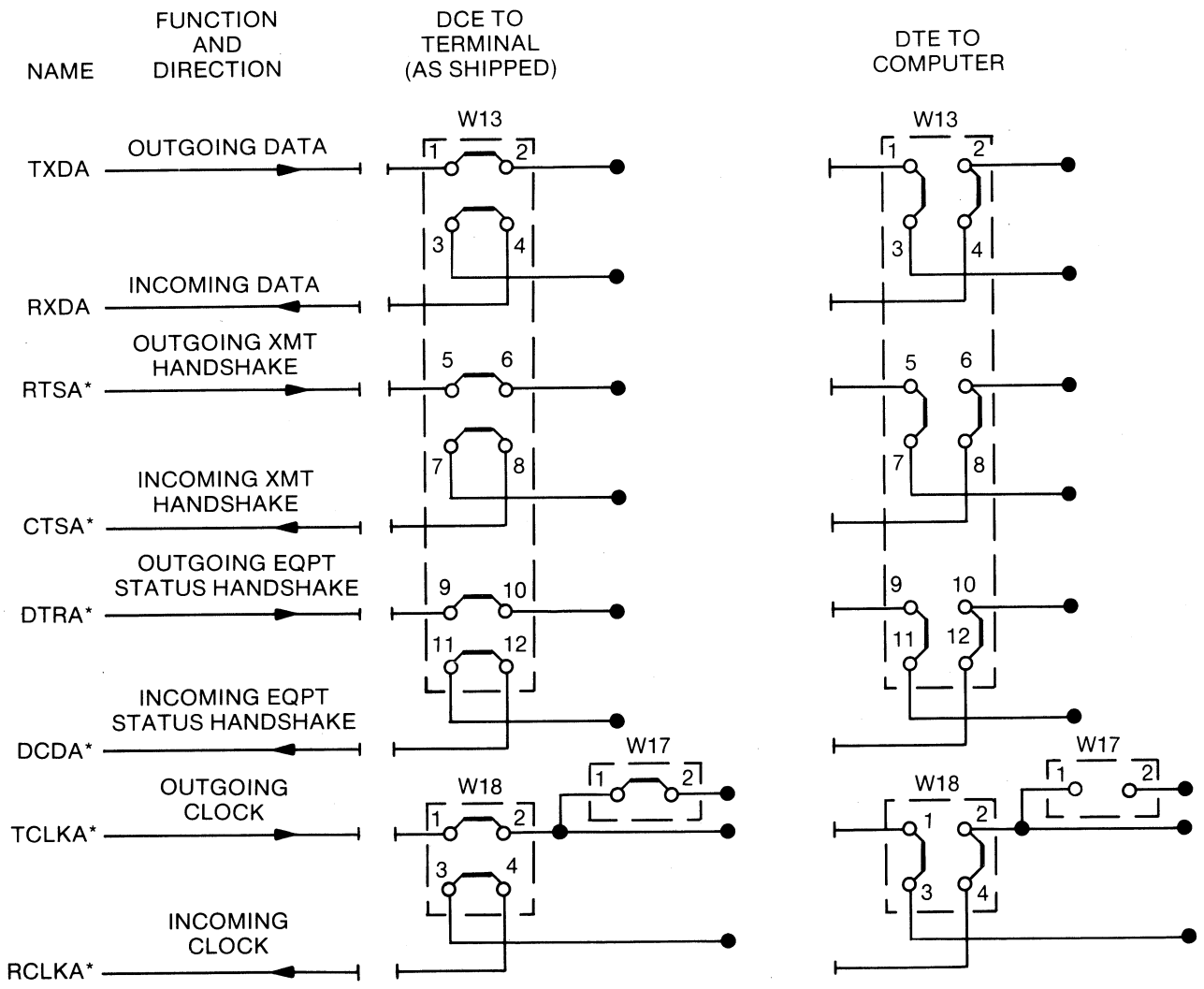


Figure 2-22. RS-232-C Serial Port DCE/DTE and DTE Clock Jumpers

DTE TRANSMIT CLOCK SOURCE, JUMPER W17 (Figure 2-23)

Some Data Terminal Equipment requires that the Data Communications Equipment connected to it provide the DTE transmit clock. The 7863 RS-232-C channel, when configured as DCE, can provide this signal (RCLK2, circuit DB on the RS-232-C cable) if a shorting connector is installed in Jumper W17, pins 1-2. When the RS-232-C channel is configured as DTE or if this clock signal is not required, the shorting connector should be removed.

DESCRIPTION	PINS WITH SHORTING CONNECTORS INSTALLED
RCLK2 (Transmit clock for external DTE) driven by RCLK 1	1-2**
RCLK2 not driven	None

**As shipped

Figure 2-23. RS-232-C External DTE Clock, Jumper W17

Section 3
7863-3 INSTALLATION

INTRODUCTION

This section describes the procedures for installing one or more 7863-3 cards in an STD BUS system and the jumper options that must be considered prior to installation.

POWER SUPPLY

CAUTION

To prevent possible damage to your STD BUS system, make sure power is off before inserting a card into the card rack or before removing a card from the card rack.

CAUTION

To guarantee proper operation of the 7863-3 card, LOGIC GND and AUX GND must be connected at the power supply.

Pro-Log power supplies are shipped with both GNDs tied together and therefore meet this requirement. If a source other than Pro-Log is used for power supplies, care must be taken to ensure this requirement is met.

In selecting a power supply for an STD BUS system that uses the 7863-3 card, make sure that all electrical requirements of the 7863-3 as specified in Section 1 (Electrical Specifications) are met. Each 7863-3can draw more than 2.4 amps of current from the 5V supply (Vcc) with memory installed. The power supply selected, the power distribution cables and connectors, and the STD BUS backplane must be able to supply the required current to each card while maintaining Vcc within the ±5% tolerance specified.

As shipped, the 7863-3 s held in a reset state until 60 milliseconds after Vcc reaches 4.75V minimum during power-up, ensuring that the card is operated within specifications. Due to component tolerances in the reset circuitry, the actual level at which reset is released and operation begins may be as high as 4.90V. Voltage drops in cabling, connectors, and the STD BUS backplane may require that the power supply be adjusted before reset is released. See the following information on Reset, Jumper W5, for more information.

PHYSICAL INSTALLATION

The 7863-3 operates as part of an STD BUS system. A 7863-3 can be inserted into any slot in the STD BUS card rack, although some systems require certain physical arrangements of cards in the card rack. These arrangements are discussed in detail under "Priority Chain." When installed, the card ejector tab must be at the top as shown in Figure 3-1. If access to the card is needed for testing or servicing, a card extender such as the Pro-Log 7911 can be used. Card extenders tend to degrade STD BUS signals, which may cause problems with dynamic RAM or other STD cards with stringent bus timing or noise requirements. The use of more than one or two extenders may cause problems in any system.

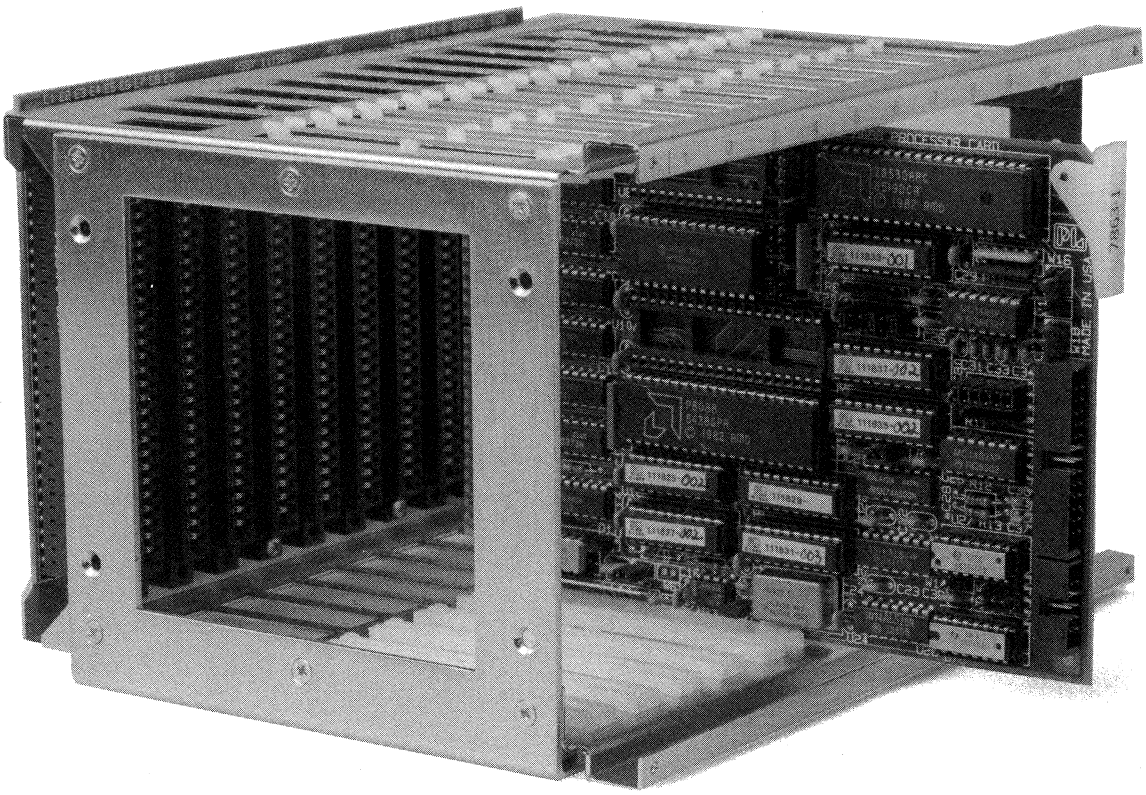


Figure 3-1. Installation

Priority Chain

The 7863-3 does not drive PCO, pin 51 on the STD BUS. Therefore, it is important not to place the 7863-3 in a card slot within a group of STD cards using the priority chain (such as I/O cards that use the chain for interrupt prioritization).

The priority chain lines on the STD BUS (PCO, pin 51, and PCI, pin 52) are used for serial interrupt or bus control. Two bus pins are allocated to the chain, which requires logic on-card to implement the serial priority function (or pass the priority chain through by connecting PCI to PCO onboard.

CARD LAYOUT AND JUMPERS

The 7863-3 has onboard jumpers to configure the card to meet system requirements. The jumper configurations consist of wire jumpers and sockets, as shown in Figure 3-2. The wire jumpers can be removed and installed without the use of soldering equipment. Figure 3-3 lists sources for the shorting connectors.

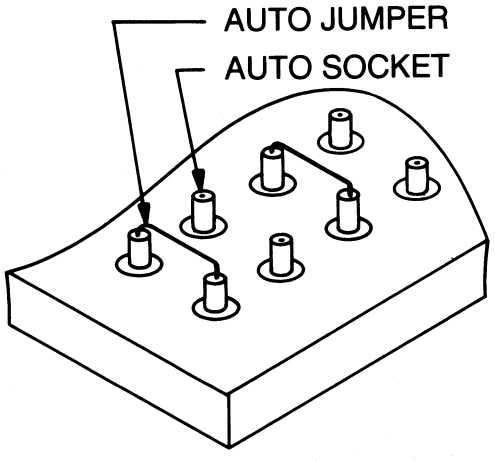


Figure 3-2. Jumper Configuration

SOURCE	PART NUMBER
Fancort Industries, Inc.	J 0.100X0.250 B 24
Squires Electronics, Inc.	J 0.100X0.250 B 24

Figure 3-3. Sources for Shorting Jumpers

U16 PAL Assembly Installation

PAL assembly 112337-xxx is installed in socket U16 for the 7863-3 multifunction card only.

Physical Layout and Jumper Options

Figure 3-4 on the following pages provides a reference for the location and a brief description of each jumper on the 7863-3. The discussion following Figure 3-4 divides the jumper options into functional groups: Reset, Memory and I/O, Interrupt, RS-422-A Serial Port, and RS-232-C Serial Port.

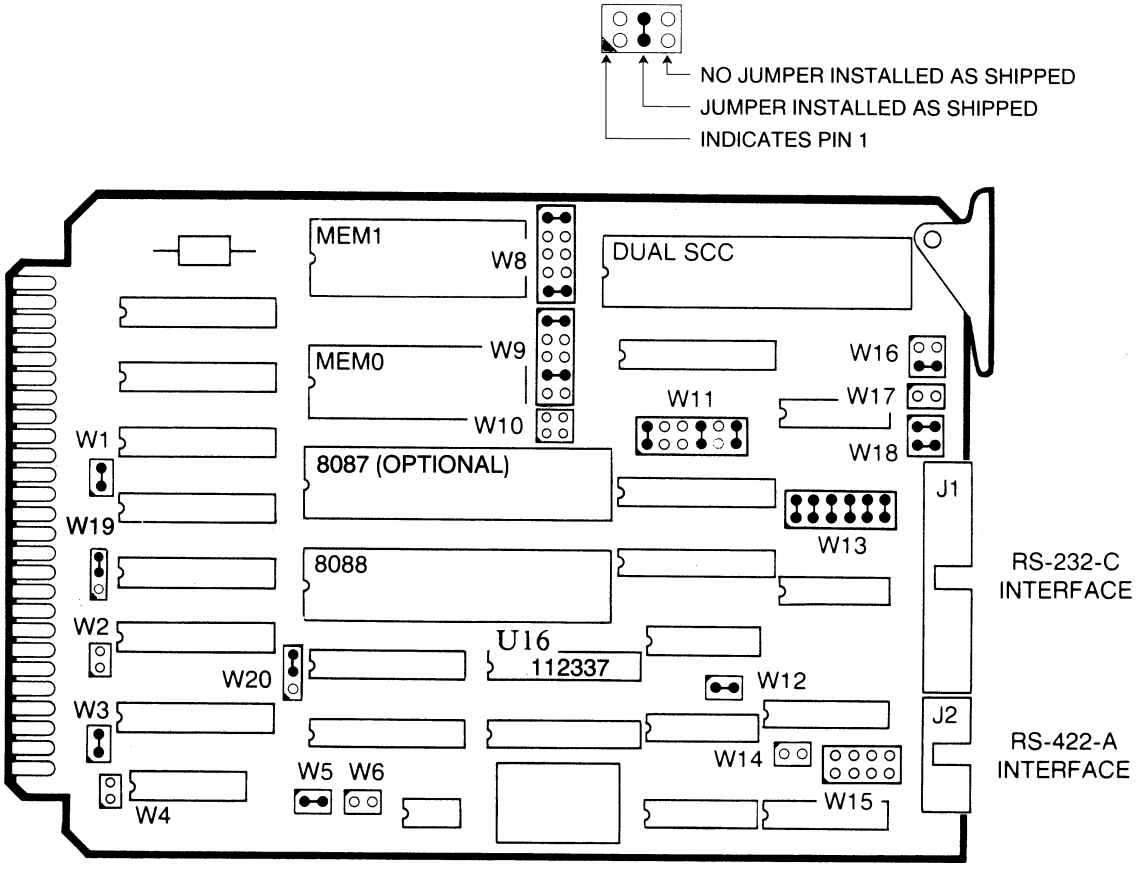


Figure 3-4. Physical Layout and Jumper Options

JUMPER GROUP	JUMPER INSTALLED	JUMPER REMOVED	DESCRIPTION	AS SHIPPED	SEE PAGE
W1			IOEXP DRIVE ENABLE		
	1-2	—	• IOEXP grounded	*	2-9
	—	1-2	• IOEXP driven by another card		
W2	—	—	Reserved for Multimaster Configuration	—	—
W3			BUS INTRQ* ENABLE		
	1-2	—	• Enable	*	2-10
	—	1-2	• Disable		
W4	—	—	Reserved for Multimaster Configuration	—	—
W5			RESET VOLTAGE SELECT		
	1-2	—	• Power-on reset level 4.75V min.	*	2-8
	—	1-2	• Power-on reset level 4.75V max.		
W6			PRO-LOG USE ONLY		
W8			MEM 1 CONFIGURATION		
	1-2, 9-10	3-4, 5-6, 7-8	• 8K ROM		
	1-2, 9-10	3-4, 5-6, 7-8	• 16K ROM		2-8
	1-2, 3-4, 9-10	5-6, 7-8	• 32K ROM	*	
	3-4, 5-6, 9-10	1-2, 7-8	• 64K ROM		
	1-2, 7-8	3-4, 5-6, 9-10	• 8K RAM		
	2-4, 7-8, 9-10	1-3, 5-6	• 32K RAM		
W9			MEM 0 CONFIGURATION		
	1-2, 9-10	3-4, 5-6, 7-8	• 8K ROM		
	1-2, 9-10	3-4, 5-6, 7-8	• 16K ROM		
	1-2, 3-4, 9-10	5-6, 7-8	• 32K ROM		
	3-4, 5-6, 9-10	1-2, 7-8	• 64K ROM		
	1-2, 7-8	3-4, 5-6, 9-10	• 8K RAM	*	2-8
	2-4, 7-8, 9-10	1-3, 5-6	• 32K RAM		
W10	—	—	Reserved for Multimaster Configuration	—	—
W11			MEMORY MAPPING		
			MEM 0: (W11, Pins 1-2, 3-4, 5-6)		
	1-2, 3-4, 5-6	—	• 00000-0FFFF (64K)		
	1-2, 3-4	5-6	• 00000-07FFF (32K)		
	1-2, 5-6	3-4	• 00000-03FFF (16K)		
	1-2	3-4, 5-6	• 00000-01FFF (8K)	*	2-9
	3-4, 5-6	1-2	• E0000-EFFFF (64K)		
	3-4	1-2, 5-6, 11-12	• F0000-F7FFF (32K)		
	5-6	1-2, 3-4, 9-10, 11-12	• FC000-FDFFF (8K)		
	—	1-2, 3-4, 5-6	• SOCKET DISABLED		
			MEM 1: (W11, Pins 7-8, 9-10, 11-12)		
	7-8, 9-10, 11-12	—	• F0000-FFFFF (64K)		
	7-8, 9-10	11-12	• F8000-FFFFF (32K)		2-9
	7-8, 11-12	9-10	• FC000-FFFFF (16K)	*	
	7-8	9-10, 11-12	• FE000-FFFFF (8K)		
	9-10, 11-12	7-8	• 10000-1FFFF (64K)		
	9-10	5-6, 7-8, 11-12	• 08000-0FFFF (32K)		
	11-12	3-4, 5-6, 7-8, 9-10	• 02000-03FFF (8K)		
	—	7-8, 9-10, 11-12	• SOCKET DISABLED		

Figure 3-4. (Continued)

JUMPER GROUP	JUMPER INSTALLED	JUMPER REMOVED	DESCRIPTION	AS SHIPPED	SEE PAGE
W12	1-2 —	— 1-2	INTRQ* VECTOR SOURCE SELECT • Bus vector during INTAK* • SCC provides vector during INTAK*	*	2-10
W13	1-2, 3-4, 5-6 7-8, 9-10, 11-12 1-3, 2-4, 5-7 6-8, 9-11, 10-12	— — — —	RS-232-C CONFIGURATION • DCE • DTE	*	2-12
W14	— 1-2	1-2 —	RS-422-A TCLK/BITBUS RTS SELECT • Transmit Clock enable • Bit Bus RTS enable	*	2-18
W15	— 1-2, 3-4	1-2, 3-4 —	RS-422-A CONFIGURATION—DATA • Separate Receive & Transmit data lines • Receive & Transmit data lines connected for multidrop	*	2-11
	— 5-6, 7-8	5-6, 7-8 —	RS-422-A CONFIGURATION—CLOCK • Separate Receive & Transmit clock lines • Transmit clock connected to receive clock for multidrop	*	2-11
W16	3-4 1-2	1-2 3-4	RS-422-A RECEIVE CLOCK SOURCE • External (from RS-422-A interface) • SCC PCLK	*	2-11
W17	1-2	—	RS-232-C DTE CLOCK • DTE transmit clock driven by 7863-3 (DCE) • DTE transmit clock not driven	*	2-13
W18	1-2, 3-4 —	— 1-3, 2-4	RS-232-C CONFIGURATION • DCE • DTE	*	2-12
W19 & 20	2-3	1-2	PRO-LOG USE ONLY	—	—

Figure 3-4. (Continued)

RESET, JUMPER W5 (Figure 3-5)

This jumper sets the voltage level to which Vcc on-card must rise before reset (and SYSRESET* on the STD BUS) is released, after a time delay. As shipped, with a shorting connector installed in Jumper W5 pins 1-2, Vcc must reach +4.75V minimum. This ensures that all devices on the 7863-3 are operating within specification (Vcc = 5V ±5%) when reset is released. Due to component tolerances, the actual voltage level at which reset is released may be as high as 4.90V. Because of voltage drops in cabling and in the STD BUS backplane and card edge connectors, Vcc on the card may never rise above 4.90V and the card will not come out of the reset state. Adjustment of the power supply or remote sensing of Vcc on the backplane may be required.

Removal of the shorting connector from Jumper W5 lowers the Vcc voltage level at which reset is released to 4.75V maximum, 4.60V minimum. Adjustment of the power supply may not then be necessary. Note that if this shorting connector is removed, reset may be released before Vcc has risen above 4.75V and the card may be operated out of specification. Pro-Log recommends that you leave the W5 shorting connector in place and adjust Vcc at the power supply if necessary to ensure correct operation.

DESCRIPTION	PINS WITH SHORTING CONNECTORS INSTALLED
Power-on Reset level is 4.75V min.	1-2**
Power-on Reset level is 4.75V max.	None

**As shipped

Figure 3-5. Reset Voltage Select, Jumper W5

Memory and I/O

The 7863-3 has two memory sockets: MEM1 and MEM0. Jumpers W8 and W9, respectively, determine what type and size of memory device can be installed in each socket and Jumper W11 determines where each device is mapped in the 8088 CPU one-megabyte address range.

The 7863-3 memory sockets can accept JEDEC-standard byte-wide ROMs and RAMs with an access time of 300 nanoseconds or less. Refer to Figure 3-6 for a list of recommended memory devices.

The only jumper that affects I/O is W1, which selects whether or not IOEXP, pin 35 on the STD BUS, is grounded onboard the 7863.

MEMORY TYPE	MEMORY SIZE	MANUFACTURER'S PART NUMBER	ACCESS TIME (nS)
EPROM	8K	Intel 2764	300 max
	16K	Intel 27128 AMD 27128	
	32K	Intel 27256 AMD 27256	
	64K	Intel 27512 AMD 27512	
RAM	8K	Hitachi HM6264 LP-15 Toshiba TC5565 PL-15	200 max
	32K	NEC D43256-15	

(1) This device not available at publication

Figure 3-6. Recommended Memory Devices

MEMORY SOCKET CONFIGURATION, JUMPERS W8 AND W9 (Figure 3-7)

These jumpers select what type of device is to be installed in sockets MEM1 and MEM0. Each socket can be independently configured to accept 8K, 16K, 32K, or 64K ROMs; or 8K or 32K RAMs. Jumper W8 configures socket MEM1 and W9 configures socket MEM0.

DESCRIPTION	PINS WITH SHORTING CONNECTORS INSTALLED
8K ROM	1-2
16K ROM	1-2, 9-10[1]
32K ROM	1-2, 3-4, 9-10
64K ROM	3-4, 5-6, 9-10
8K RAM	1-2, 7-8[2]
32K RAM	2-4, 7-8, 9-10

[1]As shipped, W13 (MEM 1)
[2]As shipped, W14 (MEM 0)

Figure 3-7. Memory Socket Configuration Jumpers W8 (MEM1) and W9 (MEM0)

MEMORY MAPPING, JUMPER W11 (Figure 3-8)

Jumper W11 determines the mapping of the memory devices installed in sockets MEM1 and MEM0. Shorting connectors installed in pins 1 through 6 determine the mapping of the device in MEM0; those installed in pins 7 through 12 determine the mapping of the device in MEM1. There are eight mapping options for each socket. If the socket is not to be used (no memory device is to be installed), the last option should be selected disabling the socket and removing it from the memory map.

To prevent the selection of two options that map both sockets into the same address range, certain combinations of options for the two sockets cannot be used. For example, if MEM0 Option 5 is selected, then MEM1 Options 0, 2, 4, and 6 cannot be used. (MEM0 Option 5 requires that no shorting connector be installed in W11 pins 11-12, while MEM1 Options 0, 2, 4, and 6 require the installation of this jumper.) Unusable combinations occur only when Option 5 or 6 is selected for either socket; these options require that certain shorting connectors not be installed.

For the most efficient use of memory space, the selected mapping option should use a block of memory of the same size as the device to be installed in the socket. For example, Option 2 for socket MEM0 maps the socket at the address block 00000-03FFF hex, a 16K block of memory; a 16K memory device should be used. If an 8K device is installed, it will be redundantly addressed in the upper half of the 16K block. If a 32K device is installed, the upper 16K of the device cannot be accessed. Additionally, two options should never be selected that map both sockets into the same address block.

DESCRIPTION			PINS WITH SHORTING CONNECTORS INSTALLED	
SOCKET	OPTION	ADDRESS RANGE	PINS 1-6	PINS 7-12
MEM 0:	0	00000-0FFFF (64K BLOCK)	1-2,3-4,5-6	DON'T CARE
	1	00000-07FFF (32K BLOCK)	1-2,3-4	
	2	00000-03FFF (16K BLOCK)	1-2,5-6	
	3	00000-01FFF (8K BLOCK)**	1-2	
	4	E0000-EFFFF (64K BLOCK)	3-4,5-6	
	5	F0000-F7FFF (32K BLOCK)	3-4	(No connector in 11-12)
	6	FC000-FDFFF (8K BLOCK)	5-6	(No connector in 9-10, 11-12)
	7	SOCKET DISABLED	NONE	DON'T CARE
MEM 1:	0	F0000-FFFFF (64K BLOCK)	DON'T CARE	7-8,9-10,11-12
	1	F8000-FFFFF (32K BLOCK)		7-8,9-10
	2	FC000-FFFFF (16K BLOCK)**		7-8,11-12
	3	FE000-FFFFF (8K BLOCK)		7-8
	4	10000-1FFFF (64K BLOCK)		9-10,11-12
	5	08000-0FFFF (32K BLOCK)	(No connector in 5-6)	9-10
	6	02000-03FFF (8K BLOCK)	(No connector in 3-4, 5-6)	11-12
	7	SOCKET DISABLED	DON'T CARE	NONE

**As shipped

Figure 3-8. Onboard Memory Mapping, Jumper W11

IOEXP DRIVE, JUMPER W1 (Figure 3-9)

If a shorting connector is installed in Jumper W1 pins 1-2, IOEXP (pin 35 on the STD BUS) is connected to ground. If the connector is removed, IOEXP is not connected on the 7863-3 and can be driven by some other card in the system if necessary. IOEXP is defined in the 8088 Bus Specification as an enable line (low-level active) for I/O cards that decode an 8-bit I/O address instead of the 8088's full 16-bit address.

DESCRIPTION	PINS WITH SHORTING CONNECTORS INSTALLED
IOEXP grounded	1-2*
IOEXP driven by another card	NONE

**As shipped

Figure 3-9. IOEXP Drive Enable, Jumper W1

Interrupt Options

Jumpers W3 and W12 determine the response of the 7863-3 to maskable interrupt requests from the STD BUS (INTRQ*, pin 44).

INTRQ ENABLE, JUMPER W3 (Figure 3-10)*

If a shorting connector is installed in Jumper W3 pins 1-2, the 7863-3 can respond to INTRQ* on the STD BUS. If the connector is removed, the 7863-3 does not respond to INTRQ*.

DESCRIPTION	PINS WITH SHORTING CONNECTORS INSTALLED
This card responds to STD BUS INTRQ*	1-2**
Does not respond to STD BUS INTRQ*	NONE

**As shipped

Figure 3-10. INTRQ* Enable, Jumper W3

INTRQ VECTOR SOURCE SELECT, JUMPER W12 (Figure 3-11)*

The 8088 CPU responds to a maskable interrupt request by executing an interrupt acknowledge cycle during which an 8-bit vector is read in on the processor data bus. This vector is normally provided by the interrupting device and is used by the processor to locate the correct interrupt service routine. If a shorting connector is installed in Jumper W12 pins 1-2, the 8088 executes an interrupt acknowledge cycle and reads a vector (which must be supplied by the interrupting device) from the STD data bus in response to an STD BUS INTRQ*.

When the shorting connector is removed from Jumper W12, INTRQ* can no longer directly cause an 8088 maskable interrupt. However, INTRQ* is permanently connected (through a buffer) to an unused handshake line input of the Serial Communications Controller (SCC). The SCC can be programmed to generate an onboard interrupt in response to transitions on that handshake line and it provides a vector on-card when the interrupt is acknowledged.

DESCRIPTION	PINS WITH SHORTING CONNECTORS INSTALLED
Vector read from STD BUS during INTAK*	1-2**
SCC provides vector during INTAK*	NONE

**As shipped

Figure 3-11. INTRQ* Vector Source Select, Jumper W12

RS-422-A Serial Communications Channel

Jumpers W14, W15, and W16 affect the RS-422-A interface, which is connected to channel B of the SCC. The pinout of the RS-422-A connector eliminates the need for DCE/DTE configuration jumpers and preserves the differential signals on twisted pairs. Figure 3-12 shows how installation of the RS-422-A cable configures the channel as either DCE or DTE. For further details, see Pro-Log Application Note #140.

Termination for the RS-422-A receive data and receive clock signals is provided by 100-ohm resistors R12 and R13. If a different resistance value is required to match line impedance, these resistors should be replaced. In multidrop applications, termination is necessary only at the two ends of the signal line. All other termination resistors should be removed since they will appear in parallel and reduce the signal.

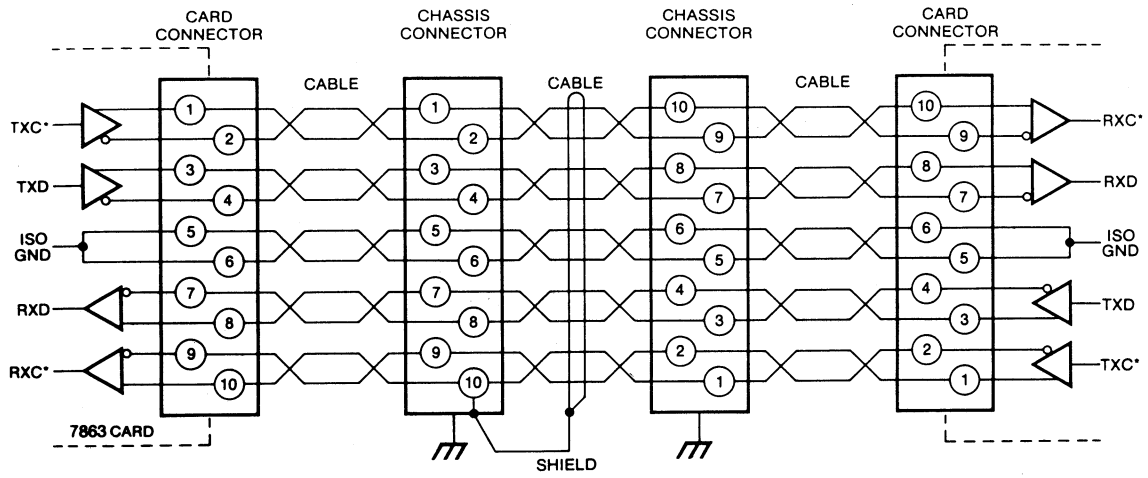


Figure 3-12. RS-422-A Synchronous System Interconnection

MULTIDROP CONFIGURATION, JUMPER W15 (Figure 3-13)

The 7863-3 RS-422-A interface meets RS-485 standards which allow up to 32 driver/receiver pairs to be placed on a single differential line. To configure the RS-422-A port for multidrop, shorting connectors should be installed in Jumper W15, pins 1-2 and 3-4, to connect the receive and transmit data lines. For synchronous operation, shorting connectors should also be installed in pins 5-6 and 7-8 to connect the receive and transmit clock lines. For applications other than multidrop, all shorting connectors should be removed.

DESCRIPTION	PINS WITH SHORTING CONNECTORS INSTALLED
Separate data and clock signals	None**
Asynchronous multidrop configuration: RxD and TxD connected	1-2,3-4
Synchronous multidrop configuration: RxD and TxD connected, RxC and TxC connected	1-2, 3-4, 5-6, 7-8

Figure 3-13. RS-422-A Configuration, Jumper W15

RS-422-A RECEIVE CLOCK SOURCE, JUMPER W16 (Figure 3-14)

Normally the source for the RS-422-A channel receive clock is the signal RCLK from

the RS-422-A connector. This option is selected by installing a shorting connector in Jumper W16, pins 3-4. In self-clocking synchronous modes, it is possible to increase the maximum possible data rate by a factor of four (to 307.2K baud) by bypassing the SCC internal baud rate generator and using the SCC clock input (PCLK, pin 20 on the SCC) as the receive clock source. (The receive clock is an input to the internal phase-locked loop in self-clocking modes.) Moving the shorting connector installed in Jumper W16 from pins 3-4 to pins 1-2 connects PCLK to the receive clock input for this purpose.

DESCRIPTION	PINS WITH SHORTING CONNECTORS INSTALLED
RCLK driven from RS-422-A interface	3-4**
RCLK driven by SCC PCLK signal	1-2

**As shipped

Figure 3-14. RS-422-A Receive Clock Source, Jumper W16

BITBUS™ TRANSCEIVER CONTROL ENABLE, JUMPER W14 (Figure 3-15)

When a shorting connector is inserted in Jumper W14 pins 1-2, the RS-422-A transmit clock lines TCLK+ and TCLK- can be used as the BITBUS Request To Send signal (RTS). This signal is used for transceiver control in a self-clocked BITBUS interconnect with repeaters. The BITBUS Interconnect Specification is available in the Distributed Control Modules Databook from Intel Corporation.

™BITBUS is a registered trademark of Intel Corporation.

DESCRIPTION	PINS WITH SHORTING CONNECTORS INSTALLED
Normal RS-422-A Transmit Clock operation	None**
RS-422-A TCLK lines used for BITBUS transceiver control (RTS signal)	1-2

**As shipped

Figure 3-15. RS-422-A TCLK/RTS Select, Jumper W14

RS-232-C Serial Communications Channel

The RS-232-C interface is connected to channel A of the Serial Communications Controller.

RS-232-C CONFIGURATION, JUMPERS W13 AND W18 (Figure 3-16)

These jumpers select whether the channel operates as data communications equipment (DCE) or data terminal equipment (DTE). The orientation of the shorting connectors installed in Jumpers W13 and W18 determine which mode of operation is selected (as shown in Figure 3-17). As shipped, the channel is configured as DCE, since it will be connected to a terminal in many applications.

DESCRIPTION	PINS WITH SHORTING CONNECTORS INSTALLED	
	W13	W18
Configured as DCE	1-2, 3-4, 5-6 7-8, 9-10, 11-12**	1-2, 3-4**
Configured as DTE	1-3, 2-4, 5-7 6-8, 9-11, 10-12	1-3, 2-4

**As shipped

Figure 3-16. RS-232-C Configuration, Jumpers W13 and W18

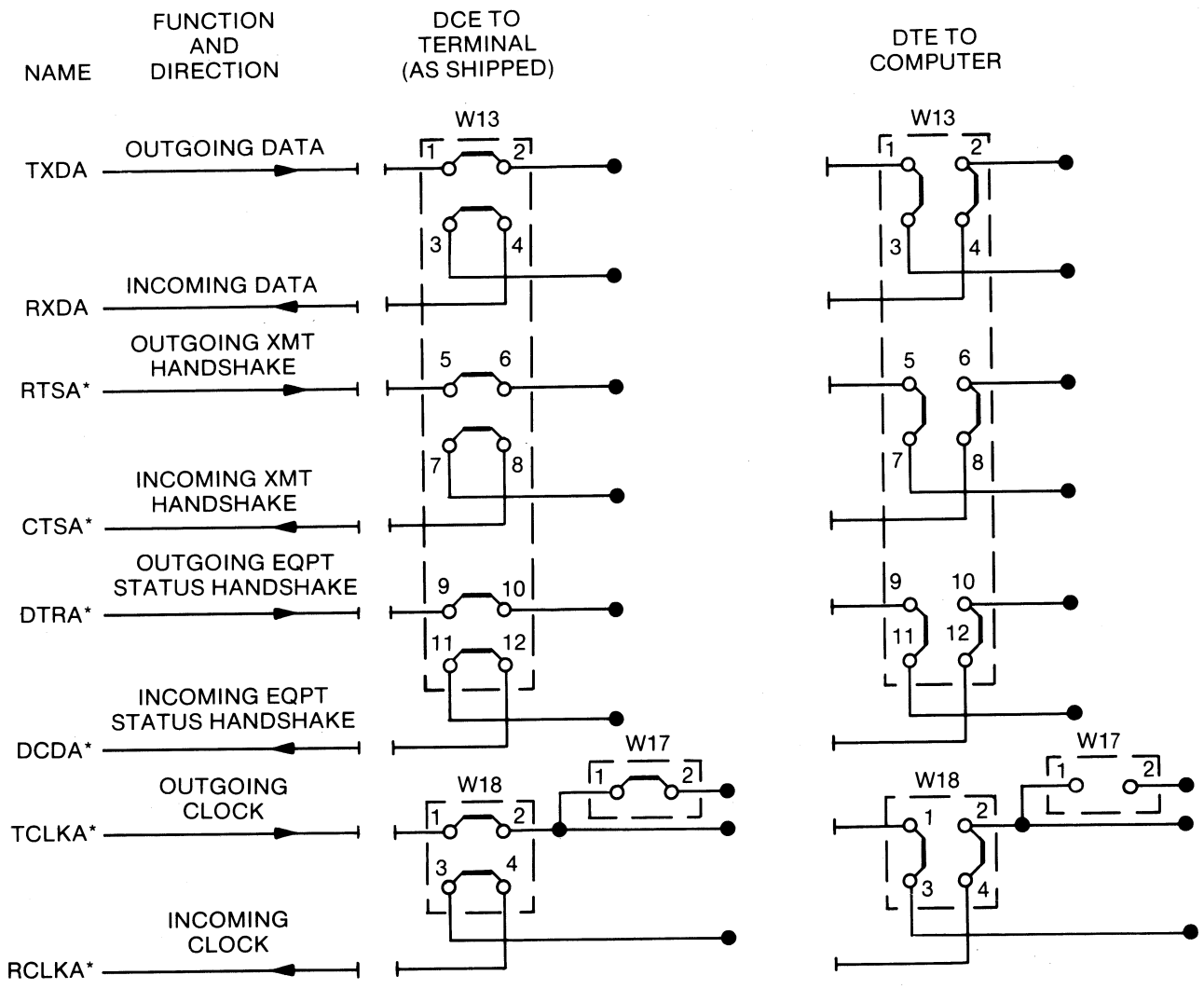


Figure 3-17. RS-232-C Serial Port DCE/DTE and DTE Clock Jumpers

DTE TRANSMIT CLOCK SOURCE, JUMPER W17 (Figure 3-18)

Some Data Terminal Equipment requires that the Data Communications Equipment connected to it provide the DTE transmit clock. The 7863-3 RS-232-C channel, when configured as DCE, can provide this signal (RCLK2, circuit DB on the RS-232-C cable) if a shorting connector is installed in Jumper W17, pins 1-2. When the RS-232-C channel is configured as DTE or if this clock signal is not required, the shorting connector should be removed.

DESCRIPTION	PINS WITH SHORTING CONNECTORS INSTALLED
RCLK2 (Transmit clock for external DTE) driven by RCLK 1	1-2**
RCLK2 not driven	None

**As shipped

Figure 3-18. RS-232-C External DTE Clock, Jumper W17

SECTION 4
7863-1 OPERATION AND PROGRAMMING

INTRODUCTION

This section includes a general discussion of multimaster STD BUS system operation, the operation of the 7863-1 Multimaster CPU card, and the programming required to initialize and use the 7863 onboard functions. These functions include the reset circuit, interrupts, and memory and I/O addressing.

SINGLE-CPU SYSTEMS

The 7863-1 can be used as the sole CPU card in a system in the same way that STD BUS systems have always been implemented in the past -- with only one CPU card in the card rack. Certain jumpers on the card should be configured for single-CPU system operation as described in Section 2 under *Multimaster Configuration Jumpers.* Operation and programming are the same as described in the following except that bus arbitration is obviously not required; the 7863 always controls the STD BUS (unless DMA operations are used) and wait states for bus arbitration are never needed for STD BUS memory or I/O access.

MULTIMASTER STD BUS SYSTEMS

The 7863-1 has the required onboard bus arbiter that allows its use in multimaster STD BUS systems. Up to sixteen 7863-1 or other Pro-Log multimaster CPU cards can be installed in the same card rack, each executing its own program independently while sharing common bus resources. These common or "global" resources include any memory or I/O not located on a multimaster card.

Bus arbitration is done entirely in hardware and is essentially transparent to the program being executed by each of the multimasters. An understanding of the arbitration process can be helpful to the user in setting up an efficient bus protocol for the multimaster system. This protocol is defined by user-provided software and allows the multimaster cards to communicate and to coordinate tasks.

Bus Arbitration

When a multimaster card is executing programs from or otherwise accessing its own onboard memory, or using onboard I/O, it does not need control of the STD BUS. Control of the STD BUS is only required to access global memory or I/O and only one multimaster can have control of the STD BUS at any one time to access these global resources. Multimaster cards usually have significant amounts of onboard memory and I/O (the 7863 can have up to 128K of memory, two serial ports, and the optional 8087 Math Coprocessor onboard) so that each can perform complex system functions without the need to continually access global memory.

When a multimaster executes a memory or I/O cycle that requires access to off-card (global) memory or I/O and does not at that time have control of the STD BUS, its bus arbiter holds the CPU in a wait state until it can gain access to the bus. The length of time that the wait state is held, i.e., until any given multimaster can gain access to the STD BUS and continue program execution, depends on several factors.

If no other multimaster is accessing the bus, the multimaster requesting bus access will obtain the bus in two CNTRL* clock cycles. If another multimaster is currently using the bus, that multimaster relinquishes the bus at the end of its current machine cycle and, after three CNTRL* clock cycles, the requesting multimaster has control of the STD BUS, the wait state is released, and program execution continues with the global memory or I/O operation that started the bus arbitration process.

Simultaneous bus access requirements from more than one multimaster are prioritized by the bus arbiters themselves, using the STD BUS priority chain. Each multimaster (starting with highest priority or rightmost) has control of the bus for one memory or I/O cycle and then relinquishes the bus to the next lower priority multimaster. If (n) multimasters request access simultaneously,

the longest that the lowest priority multimaster will be held in a wait state before being released to execute one bus memory or I/O cycle is (n-1) 8088 machine cycles plus (n) bus arbitration cycles (plus the time it takes for the multimaster originally controlling the bus to complete its cycle in progress). Any other multimaster attempting to gain access to the bus will be held in a wait state until all the multimasters which simultaneously requested bus access have had access to the bus for one cycle.

With this bus arbitration scheme, it is likely that any time a multimaster tries to access global memory or I/O it will first be held in a wait state for some period of the time. If fast program execution is necessary, the number of bus accesses should be minimized. Generally, programs being executed by a multimaster should reside in onboard memory (either RAM or ROM), since execution of programs from global memory would require a bus access to read each instruction. If more than one multimaster were executing programs from global memory, speed of program execution would be greatly reduced since each would have to alternate fetching instructions with the other multimasters with the added delay of a bus arbitration cycle each time control of the bus changed hands.

Bus Lock

Multimasters have the capability of "locking" the bus, i.e., retaining control of the bus instead of relinquishing it at the end of any machine cycle during which another multimaster requires bus access. A multimaster can lock the bus either with software (by inserting a lock prefix to an instruction in the program) or with hardware (by setting an output port bit). The bus will be locked when the multimaster implementing a lock mode gains control of the bus through the normal bus arbitration process.

Any multimaster attempting a bus access while the bus is locked by another multimaster will be held in a wait state until the lock is removed. If the bus has been locked with the software lock mode, it remains locked until the multimaster which locked it has completed execution of the instruction with the lock prefix. If the bus has been locked with hardware, it remains locked until the multimaster that locked the bus clears its lock output port bit.

Any multimaster can implement Software Lock or Hardware Lock modes as required. The highest priority multimaster (the rightmost in the card rack) can also use a slightly different type of hardware lock called DMA Lock. This mode is the same as Hardware Lock except that while DMA Lock mode is active, the STD BUS signals BUSRQ* and BUSAK* can be used in the usual way by DMA controllers on non-multimaster STD cards. Traditional STD BUS DMA operations are possible only when DMA Lock mode is active since multimaster bus arbiters use BUSRQ*/BUSAK* in a different way from normal DMA controllers (BUSAK* active indicates that the bus is being used by a multimaster instead of indicating that the bus is available for use by the device that supplied BUSRQ*). This restriction on the use of DMA controllers is not usually a significant limitation in a multimaster STD BUS system since each multimaster always has direct access to all global system memory and I/O. One multimaster can, in effect, handle DMA tasks for another.

As noted above, any multimaster attempting a bus access while a lock mode is active is held in a wait state until the lock mode has been cleared by the multimaster currently using the bus. The use of lock modes should be carefully considered by the program designer who must ensure that no time-critical function is affected by a bus lock. It may be desirable to limit the duration of a Hardware or DMA lock, either in software or with timer-generated interrupts, so that other multimasters are not denied access to the bus (and held in a wait state) for too long. The 7863-1 SCC baud rate generator can be used as a counter/timer for this purpose.

Multimaster System Configurations

The interface between multimasters is well defined from a hardware standpoint by the bus arbitration scheme, yet flexible enough to support a variety of different system configurations. The way multimasters operate together in the system is mainly dependent upon software written by the user.

A multimaster system can be set up in a conventional master-slave configuration in which one multimaster is designated (in software) as the master and all others as slaves. The master off-loads specific tasks to specific slaves: one slave may be used exclusively for communications, another for arithmetic computation, another for data acquisition, and so on. Writing the software for such a system might be simpler than for an equivalent single-CPU system (since each multimaster has only a portion of the overall system task to perform) and throughput could be greatly increased since all lower-level tasks would be executed in parallel by the slave multimasters.

In another possible system configuration, two or more multimasters might run entirely unrelated programs and treat other cards on the STD BUS as shared resources. These might include other multimasters as well as memory and/or I/O cards. In such a system, several "resource" multimasters might be used for exactly the same function: they could perform computations for each of the "masters" in the system as needed since each can have its own onboard 8087 math coprocessor.

BUS PROTOCOL
(Communication Between Multimasters)

The hardware interface is very flexible and places few restrictions on system configuration. Therefore, the bus protocol that defines the way in which multimasters communicate on the bus can be designed by the user and implemented in software.

Shared Memory (Mailboxes)

Communication between multimasters in an STD BUS system is usually through global memory (memory not located on a multimaster but on a separate memory card). Most of this global memory is directly addressable by all multimasters (see the subsection on memory and I/O addressing following). Certain areas in global read/write (RAM) memory can be designated by the program designer as "mailboxes" where commands, status information, and data can be written by one multimaster and read by another.

The 8088 XCHG (exchange) instruction can be used to pass commands and/or status information between two multimasters. This instruction switches the contents of source and destination operands. When this instruction is used for multimaster communication, one operand is a register in the multimaster's CPU and the other operand is a global read/write memory mailbox location. The XCHG instruction can be used with the LOCK prefix to implement the Software Lock mode described above. This ensures that no other multimaster can access (change) the memory location operand while the exchange is actually taking place.

Figures 4-1 through 4-3 show some examples of how the XCHG instruction with LOCK prefix can be used in multimaster communications. In Figure 4-1, a byte labeled "CNTRL1" has previously been defined in global RAM memory and serves as an access control flag for some global bus resource (an I/O device, a block of memory, etc.). If CNTRL1=0, the resource is available for use; if CNTRL1=1, another multimaster is using the resource. The program shown in Figure 4-1 would be executed by any multimaster requiring use of the resource. The multimaster would wait in the loop at the beginning of the code until CNTRL1=0. The use of the LOCK prefix to the XCHG instruction prevents more than one multimaster from changing the access control flag from 0 to 1 (i.e., getting control of the resource) at one time.

```
;This is a location in global RAM:

CNTRL1      DB      Ø           ;Access control flag for global resource

;Any multimaster requiring use of the resource executes this program:

CHECK:      MOV     AL,1        ;Set AL to 1
     LOCK   XCHG    CNTRL1,AL   ;Exchange contents of AL and CNTRL1
            TEST    AL,AL       ;Set Z flag if CNTRL1 was Ø
            JNZ     CHECK       ;Retry if CNTRL1 was not Ø
             -                  ;CNTRL1 was Ø and is now 1
             -                  ;     This multimaster now has control
             -                  ;     of resource, and can continue
             -                  ;     operation
            MOV     CNTRL1,Ø    ;Operation completed - clear CNTRL1 so
                                ;     other multimasters can use resource
```

Figure 4-1. Use of XCHG Instruction with Lock Prefix

Figure 4-2 shows how two access control flags can be used to control a swing buffer for one-way communication of blocks of data between two multimasters. The buffer

is split into two halves, each having a separate access control flag. If an access control flag equals 0, the associated half of the buffer has been emptied by the receiving multimaster and the transmitting multimaster can move more data into it. If the flag equals 1, the associated half-buffer has been filled by the transmitting multimaster (using the LOCK XCHG instruction) after it has moved a block of data into an empty half-buffer. The flag is cleared by the receiving multimaster (with LOCK XCHG) after it has read the data in from a full half-buffer. The advantage of using a swing buffer is that both multimasters can be moving data simultaneously into and out of the buffer without having to wait for the other to finish its operation.

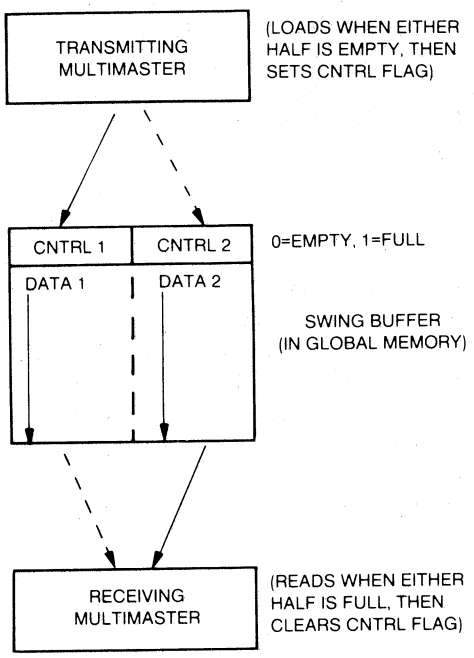


Figure 4-2. Multimaster Swing Buffer

Software Interrupts

Figure 4-3 shows how the XCHG instruction could be used in conjunction with the software interrupt instruction (INT n, where n = interrupt type number) to pass commands between multimasters. Figure 4-3A shows programs executed by one multimaster and Figure 4-3B shows programs executed by any other multimaster that needs to communicate with the first.

The programs for both multimasters define a five-byte area in global RAM. The first byte is an access control flag similar to that previously described, while the following three bytes are a subroutine (CMD_1) that will be periodically executed by the first multimaster. Three bytes must be initialized so that if the subroutine is called by the first multimaster, it will execute a locked two-byte instruction (XCHG BX,BX) and an inter-segment return (RET) while waiting for another multimaster to communicate.

When another multimaster needs to communicate with the first, it replaces the no-operation instruction (XCHG BX,BX) with an INT n instruction. The first multimaster's program executes the INT n instruction the next time the subroutine CMD_1 is called. The INT n service routine takes whatever action is predefined and then replaces the INT n with two-byte no-operation instruction and clears the access control flag, allowing another multimaster to repeat the communication procedure as needed. Using the LOCK prefix ensures that only one multimaster at a time can use this procedure.

The use of software interrupts is an efficient method of multimaster communication. Since n (the interrupt type number) can be any value between 32 and 255, many commands can be implemented by writing an appropriate interrupt service routine for each without the need for polling.

An access control flag (CNTRL2) can also be used to control access to a block of global memory which serves as the mailbox for passing data or parameters required by the specific command.

Periodic STD BUS Access

Each multimaster periodically reads or executes code from global read/write memory to see if new commands, status information, or data are being sent to it from another multimaster. The frequency of bus access for this purpose depends upon the system configuration and the particular application. In a master/slave/smart I/O system, each slave might not access its global memory mailboxes

```
;All multimasters can access this block of global RAM:

CNTRL2      DB      Ø           ;This is the flag for access control
CMD_1       PROC FAR            ;Subroutine executed by 1st multimaster
            DB      ØFØH        ;Code for a LOCK prefix instruction
INSTR       DW      ØBB87H      ;Code for a two byte NOP instruction
                                   (i.e. XCHG  BX,BX)
            RET                 ;Return from subroutine
CMD_1       ENDP                ;

;1st multimaster periodically executes this instruction:

            CALL    CMD_1       ;Execute subroutine in global RAM

;After another multimaster has moved "INT n" code to INSTR, the next CALL
;     CMD_1 will cause an INT n
;The 4-byte interrupt vector table entry at Address ØØØØ:(n*4) is the
;     offset and segment of this interrupt service routine:

INTn_SVC    PROC FAR            ;Interrupt Service Routine
             -                  ;(Instructions are inserted here which
             -                  ;     perform the actual interrupt
             -                  ;     service operations)
            MOV    INSTR,ØBB87H ;Replace INT n code with a two byte NOP instruction
                                   (XCHG  BX,BX)
            MOV    CNTRL2,Ø     ;Clear access control flag
            IRET                ;Return from interrupt n service
INTn_SVC    ENDP                ;
```

A) FIRST MULTIMASTER'S PROGRAM

```
;All multimasters can access this block of global RAM:

CNTRL2      DB      Ø           ;This is the flag for access control
CMD_1       PROC FAR            ;Subroutine executed by 1st multimaster
            DB      ØFØH        ;Code for a LOCK prefix instruction
INSTR       DW      ØBB87H      ;Code for a two byte NOP instruction (XCHG  BX,BX)
            RET                 ;Return from subroutine
CMD_1       ENDP                ;

;To cause 1st multimaster to execute an INT n (n = type number, 32 to 255),
;     another multimaster executes this program:

WAIT:       MOV     AL,1        ;Try to set access control flag
     LOCK   XCHG    CNTRL2,AL   ;
            TEST    AL,AL       ;Wait until CNTRL2 equaled Ø during XCHG
            JNZ     WAIT        ;
             -                  ;(Insert additional instructions here-
             -                  ;     set up data to be passed, etc.)
     LOCK   MOV     INSTR,nCDH  ;Move code for INT n to RAM
```

B) ANY OTHER MULTIMASTER'S WITH SOFTWARE INTERRUPTS

Figure 4-3. Multimaster Communication with Software Interrupts

until it had completed an assigned task. At that time, it would move the results (if any) and "task-completed" status into the appropriate mailboxes and then continually monitor its command mailbox, waiting for a new task to be assigned. The "master" multimaster in the same system would scan the slaves' status mailboxes, utilize the results from slave operations when available, and assign new tasks to the slaves as tasks were completed.

Interrupts

Onboard periodic interrupts on each multimaster could determine the frequency of bus access for reading command or status mailboxes. Multimasters with an onboard timer function (7864-1) that can generate an interrupt could check the appropriate mailboxes or execute a program in global memory when interrupted. The 7863's Serial Communications Controller can generate a periodic onboard interrupt with its baud rate generator that could be used for this purpose. This timed approach might be used in a true multimaster system where each multimaster might periodically check to see what bus resources were available for use.

Figure 4-4 shows how periodic interrupts could be used with the software interrupts described above. The code is similar to that in Figure 4-3, except that the subroutine in

global RAM executed by the first multimaster is now the interrupt service routine for the periodic hardware interrupt INT x. Another multimaster needing to communicate with the first executes the program shown in Figure 4-3B, causing the first multimaster to execute a software INT n at the time of the next periodic hardware interrupt.

```
;All multimasters can access this block of global RAM:

CNTRL2      DB      0           ;This is the flag for access control
INTx_SVC    PROC                ;Interrupt service routine for 1st multimaster
            DB      0F0H        ;Code for A LOCK prefix instruction
INSTR       DW      0BB87H      ;Code for a two byte NOP instruction (XCHG  BX,BX)
            IRET                ;Return from interrupt
INTx_SVC    ENDP                ;

;1st multimaster periodically gets an interrupt from hardware which
;     supplies the vector x, causing an INT x
;The 4-byte interrupt vector table entry at Address 0000:(x*4) is the
;     offset and segment of INTx_SVC, the interrupt service routine

;After another multimaster has moved "INT n" code to INSTR, the next
;     hardware INT x interrupt service will cause a software INT n
;The 4-byte interrupt vector table entry at Address 0000:(n*4) is the
;     offset and segment of this interrupt service routine:

INTn_SVC:     -                 ;(Instructions are inserted here which
              -                 ;     perform the actual interrupt
              -                 ;     service operations)
              MOV     INSTR,0BB87H;Replace INT n code with a two byte NOP
                                    instruction (XCHG  BX,BX)
              MOV     CNTRL2,0  ;Clear access control flag
              IRET              ;Return from interrupt n service
```

Figure 4-4. Communications with Periodic Interrupts

An I/O card that can generate periodic bus interrupts (INTRQ* on the STD BUS) such as Pro-Log's 7308 Counter/Timer card could perform the same function. The 7308 does not provide an 8088 interrupt vector, so each multimaster must provide its own vector on-card when responding to an STD BUS INTRQ*. Each multimaster would respond to the bus interrupt by accessing its mailboxes to see if updated commands, status, or data were available to it. If faster response to some command were required, a multimaster could use the 7308 to generate an almost immediate bus interrupt by programming another counter/timer channel to down-count from a count of one and enabling the interrupt from that channel.

OPERATION AND PROGRAMMING

Reset

The 7863-1 resets the 8088 CPU, the 8087 Math Coprocessor, and certain onboard logic (including the bus arbiter) upon entering a reset state. The STD BUS signal SYSRESET* is also driven active (low) during this time. The source of the reset can be either the onboard power-on/DC power-fail circuitry or pushbutton or system reset from the STD BUS.

POWER-ON/DC POWER-FAIL RESET

The 7863-1 uses a precision voltage reference and comparator to monitor the level of Vcc on-card. During power-on, the reset state is held for 60 milliseconds after Vcc has reached 4.75V minimum. This ensures that all components on the card are operating within specification (Vcc = 5V ±5%) when reset is released. Reset is also asserted when Vcc on-card falls below a specified level. Refer to Section 1, *Electrical Specifications*, for exact voltage levels and tolerances. Under certain conditions, adjustment of the power supply or a jumper change in the reset circuit may be required before reset will be released and the 7863 begins operation. See the jumper reset options in Section 2, *Installation.*

PUSHBUTTON RESET

The 7863-1 can be forced into a reset state with an active (low) level on the STD BUS pushbutton reset (PBRESET*) signal. PBRESET* can be either controlled by a

switch or driven with an open collector device. The signal is debounced on-card and must remain low for a minimum of 5 milliseconds to guarantee a reset.

SYSTEM RESET

An active (low) level on the STD BUS system reset signal (SYSRESET*) will also cause a 7863-1 reset. (The 7863-1 will not drive SYSRESET* in response to this reset.) This signal is not debounced and should be driven only by circuitry specifically designed for this purpose (such as an AC power-fail detect card or the reset circuitry on other multimasters).

In a multimaster system, SYSRESET* is not released until the power-on circuits on all multimasters have sensed the correct Vcc level and completed their time delays, thus assuring that all multimasters start program execution simultaneously. Similarly, if any multimaster detects a DC power-fail condition, all multimasters are reset by SYSRESET*. Refer to Section 1, *Electrical Specifications*, and Section 2, *Reset Jumper W5*, for voltage levels and other reset considerations.

When reset is released, program execution begins at physical address FFFF0 hex (code segment = FFFF hex, instruction pointer = 0000 hex).

Interrupts

Every 8088 interrupt has a type number that identifies it to the CPU. The type number is used in the 8088 vectored interrupt scheme. It is a single-byte value that the CPU internally multiplies by four to form a pointer. The pointer points to an interrupt table in segment 0000 hex containing the address of the interrupt service routine. The multiplication of the type number (vector) by four allows the complete physical address (Code Segment and Instruction Pointer, two 16-bit numbers) of the interrupt service routine to be stored in the interrupt table as four consecutive bytes.

The 7863-1 8088 CPU can respond to both internal and external (to the CPU) interrupts. Sources for external interrupts on the 7863-1 can include the STD BUS signals NMIRQ* (nonmaskable interrupt request) and the INTRQ* (interrupt request) and the onboard Serial Communications Controller's interrupt output.

Nonmaskable interrupt requests on the STD BUS NMIRQ* signal are predefined by the 8088 as Type 2 interrupts and no other vector is needed. Figure 4-5 shows how the CPU processes an NMIRQ*. A Type 2 interrupt has a higher priority than any other external interrupt and since it is nonmaskable (the CPU always responds to a Type 2 interrupt at the end of its current instruction cycle), NMIRQ* is generally used to signal the CPU of a "catastrophic" event such as the imminent loss of power or a parity error.

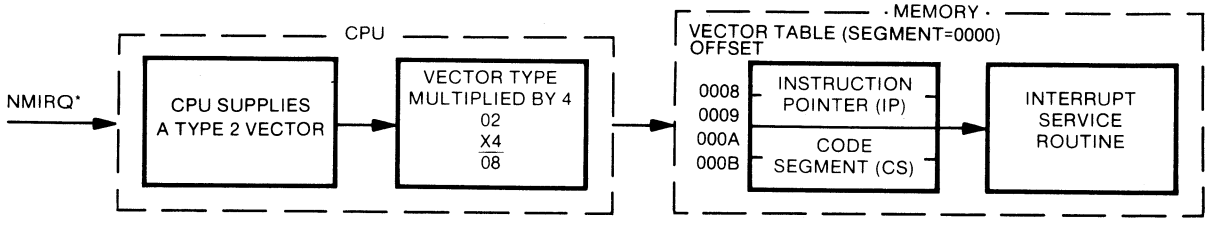


Figure 4-5. NMIRQ* Processing

Interrupt requests on the STD BUS INTRQ* signal and from the onboard SCC are not assigned a type number by the CPU. The interrupting device must provide the type number (or vector) on the CPU's data bus during the interrupt acknowledge cycle. Figure 4-6 shows how the 8088 processes these interrupts. The 7863-1 differentiates between STD BUS INTRQ* and interrupt requests from the onboard SCC; INTRQ* has a higher priority than onboard interrupts. When the 7863-1 responds to an onboard interrupt request, it is assumed that the vector will be generated on-card (by the SCC) and an STD BUS access is not attempted during the interrupt acknowledge cycle.

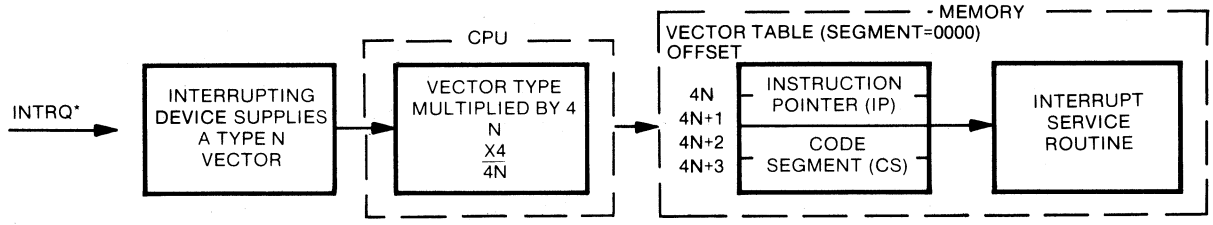


Figure 4-6. INTRQ* Processing

In multimaster systems, only one CPU card can normally respond to an STD BUS INTRQ* by gaining bus access, executing an interrupt acknowledge cycle, and reading the type number (vector) supplied by the interrupting device from the STD BUS data bus. (If more than one multimaster executed an interrupt acknowledge cycle on the STD BUS, only the highest priority would get a vector since most peripheral devices only put out a vector once when the interrupt is first acknowledged.)

If other multimasters are to respond to an STD BUS INTRQ*, they must each generate a vector onboard. This capability is provided on the 7863-1 since INTRQ* is connected (through a buffer) to an unused handshake line on the SCC. The SCC can be programmed to generate an interrupt when the handshake line changes and to provide the required on-card vector when the interrupt is acknowledged. This onboard vector capability allows the use of cards that can generate INTRQ* but not an 8088 vector in a 7863-1 system. All 7863-1 cards would respond to INTRQ* in this way. The shorting connector should be removed from Jumper W12 if this onboard INTRQ* response is to be used, so that the 7863-1 does not respond to an STD BUS INTRQ* with a bus interrupt acknowledge cycle.

If the shorting connector is removed from Jumper W3 on the 7863-1, the card will not respond to STD BUS INTRQ* at all. Refer to Interrupt Options in Section 2 for more information on maskable interrupt response.

The 8087 Math Coprocessor can generate interrupts when certain internal conditions or "exceptions" (such as stack overflow or underflow or indeterminate operations such as divide by zero) are detected. The 8087 interrupt output drives an unused handshake line input on the SCC that can be programmed to provide an on-card interrupt and vector for 8087 interrupts. SCC handshake interrupts are both rising- and falling-edge sensitive rather than level sensitive so an 8087 interrupt (or an INTRQ* "onboard" interrupt as mentioned above) causes two SCC External/Status interrupts -- one when the interrupt occurs and one when the interrupt is cleared. The interrupt service routine must first check the actual state of the handshake line by reading an SCC register to determine whether or not the interrupt needs service.

Memory Addressing

The 7863-1 8088 CPU provides a 20-bit memory address during each memory operation so the 7863-1 can direct address 2^{20} (1,048,576 or 1M) bytes of memory. The two

onboard memory sockets can be configured to support various sizes of memory devices and can be mapped at different locations within the one megabyte memory map. See Figure 2-13 for onboard memory mapping options. Generally, one memory socket is mapped at the high end of memory to support the 8088 power-on restart address (FFFF0 hex) while the other is mapped at the low end to include the 8088 interrupt vector table (00000 to 007FF hex).

In a multimaster system, the 7863-1 can address its onboard memory and continue program execution while the STD BUS is being used by another multimaster; it does not need control of the bus. The 7863-1 needs access to (and may have to arbitrate for) the STD BUS only when addressing memory that is not mapped into its onboard sockets. In addition, no other multimaster can ever access the 7863-1 onboard memory.

When the 7863-1 is addressing off-card or global memory (in a multimaster system when it has control of the STD BUS), the lower 16 address bits are output on address lines A0-A15 on the STD BUS. The upper four bits (A16-A19) are time multiplexed onto the STD data bus (bits D0-D3) at the beginning of each machine cycle. Memory cards used in 8088 STD systems latch these upper address lines on the rising edge of STD BUS control signal MCSYNC*.

MEMEX is defined in the 8088 STD BUS specification as an enable for memory cards that only decode a 16-bit memory address. No provision is made for driving MEMEX on the 7863-1 and only memory cards with the onboard latch required to decode the full 20-bit 8088 address should be used with the 7863-1 to avoid redundant addressing of memory.

Several examples of memory mapping in 7863-1 STD systems, both single-CPU and multimaster, are shown in Figure 4-7. Note that in multimaster systems, the high and low ends of the memory map are usually the onboard memory exclusively available to each multimaster and most of the memory map (always at least 792K bytes when using 7863-1s as multimasters) is available as common or global memory.

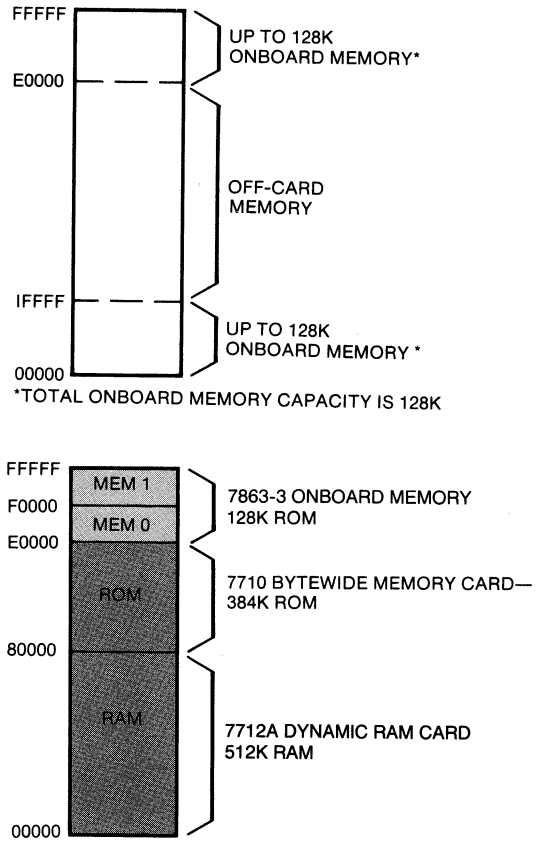


Figure 4-7. Memory Mapping Examples

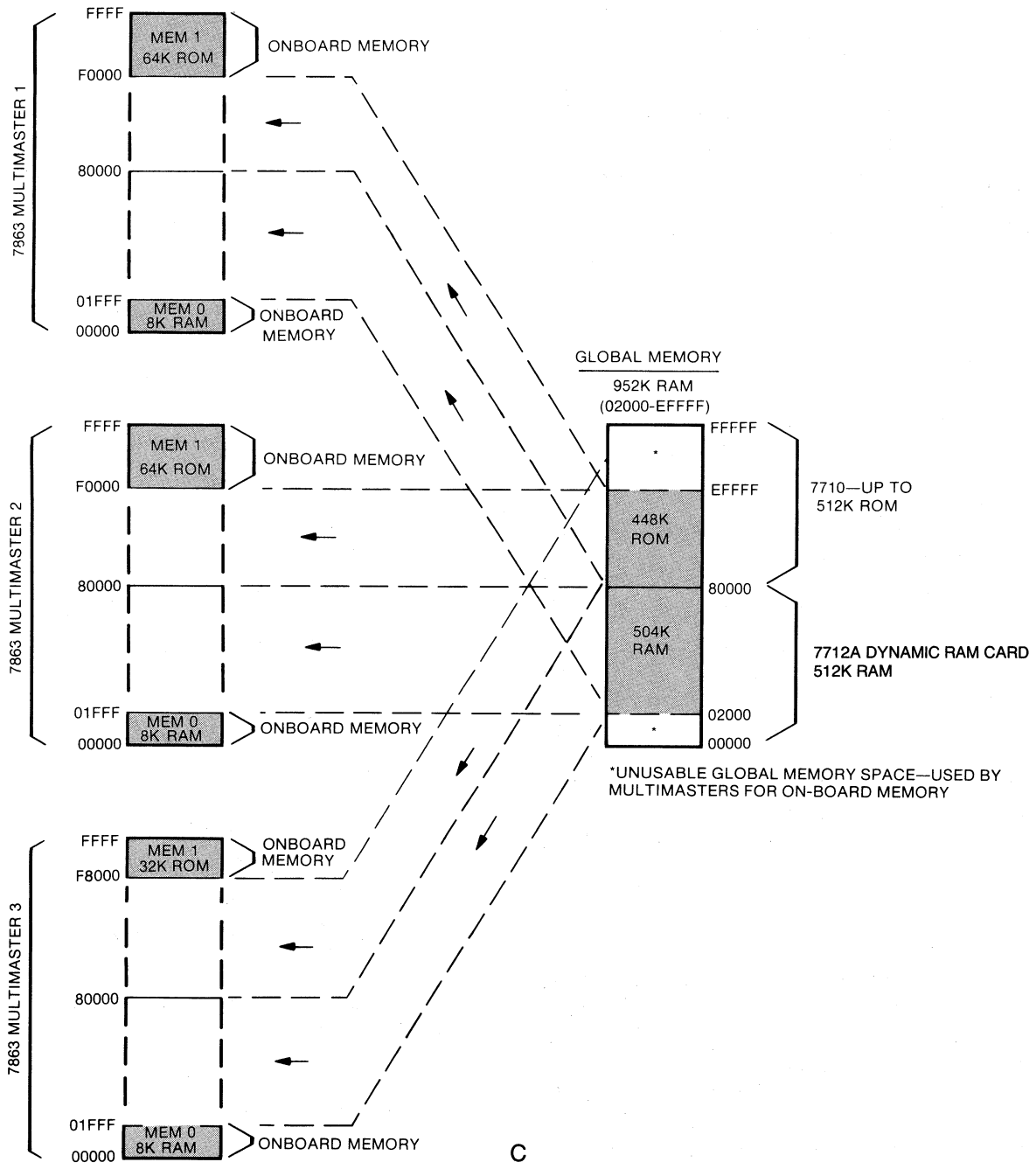


Figure 4-7. (Continued)

A block of global memory may be mapped at the same address as any multimaster's on-board memory, but it cannot be used by any multimaster in the system. (This situation is unavoidable when using high-capacity dynamic RAM cards such as the 7712A in Figure 4-7C.) When a 7863-1 currently controlling the STD BUS accesses its onboard memory, it puts out the address and control signals onto the STD BUS, but its data bus buffer is disabled. There is no bus contention with global memory mapped at the same address, but the global memory will be altered by write operations.

I/O Addressing

The 7863-1 8088 CPU provides a 16-bit I/O port address during each I/O operation and can directly address 2^{16} (65,536 or 64K) I/O port locations. As with memory, onboard I/O (the SCC and the bus lock port) can be read from or written to without having access to the STD BUS. Also like onboard memory, these onboard ports are not accessible to any other multimaster in the system. The 7863-1 onboard bus lock port is mapped at Port FFFF hex. When bit D0 is set, the lock mode selected (either Hardware or DMA Lock) is active and when bit D0 is cleared, the bus lock is released. Bits D1-D7 are not used. The SCC uses ports BFFC-BFFF hex. See Figure 1-5 for port definitions.

Due to onboard decoding limitations, the upper half of the 64K I/O address field is reserved for onboard I/O even though only Ports BFFC-BFFF and Port FFFF are actually used. The lower 32K (Ports 0000-7FFF hex) is available for common or global I/O that can be addressed by all multimasters in the system. See Figure 1-6 for 7863 I/O mapping. Note that Ports 8000-BFFB and C0000-FFFE hex should never be accessed since the onboard I/O devices will be redundantly addressed in these address ranges.

Most existing STD BUS I/O cards decode only an 8- or 10-bit I/O address since few systems need more than 256 or 1024 ports. The full port address range of these cards (00-FB for 8-bit or 000-3FB for 10-bit) is available for global I/O in a 7863-1 system. Indirect (16-bit) port addressing should be used when addressing 10-bit I/O cards to allow the card to distinguish between onboard and global ports. Eight-bit I/O cards can be directly addressed in Page 00. Eight- and ten-bit I/O cards are not usually used together in the same system since port addresses would have to be selected with great care to avoid redundant addressing ("shadow" ports). Provision is made for grounding the STD BUS IOEXP signal onboard the 7863-1 since many 8-bit address I/O cards use this signal as an enable input. Several examples of I/O port configurations in 7863-1 systems are shown in Figure 4-8.

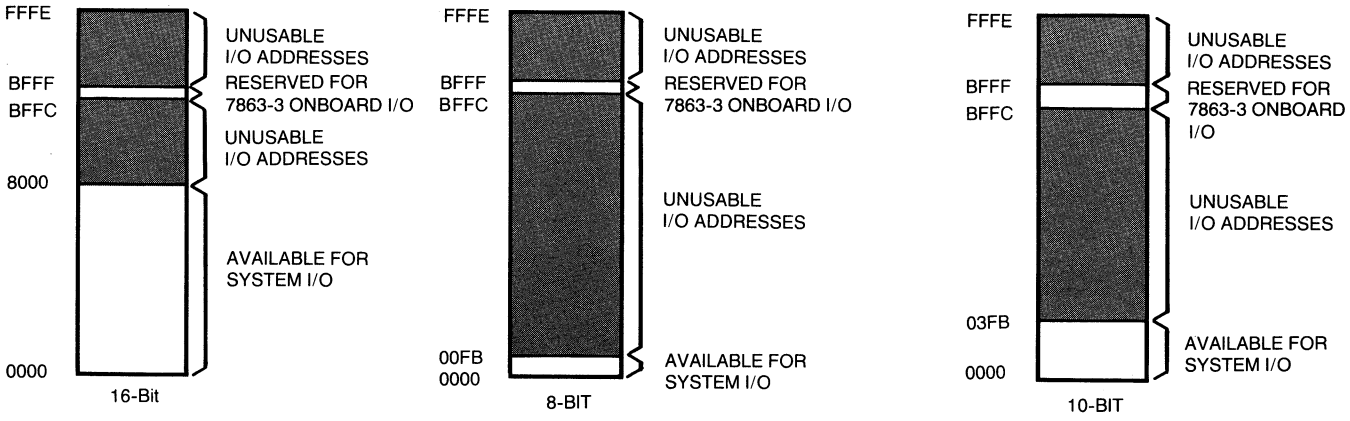
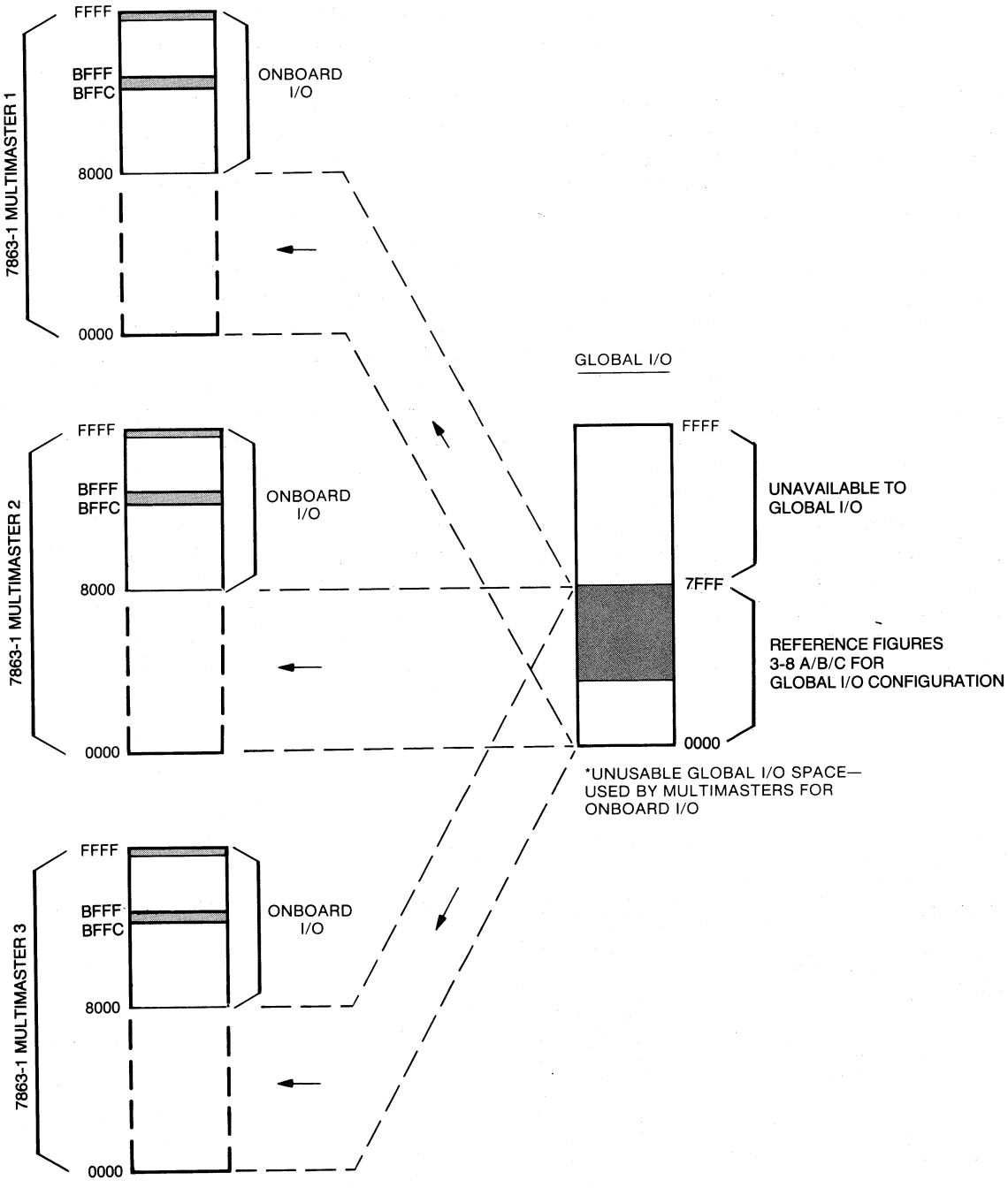


Figure 4-8. I/O Port Configuration Examples



D

Figure 4-8. (Continued)

INTRODUCTION

This section includes a general discussion of the operation of the 7863-3 Multifunction CPU card, and the programming required to initialize and use the 7863-3 onboard functions. These functions include the reset circuit, interrupts, and memory and I/O addressing. The 7863-3 is used as the sole CPU card in a system in the same way that STD BUS systems have always been implemented in the past (with only one CPU card in the card rack). The 7863-3 always controls the STD BUS unless DMA operations are used.

OPERATION AND PROGRAMMING

Reset

The 7863-3 resets the 8088 CPU, the 8087 Math Coprocessor, and certain onboard logic upon entering a reset state. The STD BUS signal SYSRESET* is also driven active (low) during this time. The source of the reset can be either the onboard power-on/DC power-fail circuitry or pushbutton or system reset from the STD BUS.

POWER-ON/DC POWER-FAIL RESET

The 7863-3 uses a precision voltage reference and comparator to monitor the level of Vcc on-card. During power-on, the reset state is held for 60 milliseconds after Vcc has reached 4.75V minimum. This ensures that all components on the card are operating within specification (Vcc = 5V ±5%) when reset is released. Reset is also asserted when Vcc on-card falls below a specified level. Refer to Section 1, *Electrical Specifications*, for exact voltage levels and tolerances. Under certain conditions, adjustment of the power supply or a jumper change in the reset circuit may be required before reset will be released and the 7863-3 begins operation. See the jumper reset options in Section 3, *Installation.*

PUSHBUTTON RESET

The 7863-3 can be forced into a reset state with an active (low) level on the STD BUS pushbutton reset (PBRESET*) signal. PBRESET* can be either controlled by a switch or driven with an open collector device. The signal is debounced on-card and must remain low for a minimum of 5 milliseconds to guarantee a reset.

SYSTEM RESET

An active (low)level on the STD BUS system reset signal (SYSRESET*) will also cause a 7863-3 reset. (The 7863-3 will not drive SYSRESET* in response to this reset.) This signal is not debounced and should be driven only by circuitry specifically designed for this purpose (such as an AC power-fail detect card or the reset circuitry on another STD card).

Refer to Section 1, *Electrical Specifications*, and Section 3, *Reset Jumper W5*, for voltage levels and other reset considerations. When reset is released, program execution begins at physical address FFFF0 hex (code segment = FFFF hex, instruction pointer = 0000 hex).

Interrupts

Every 8088 interrupt has a type number that identifies it to the CPU. The type number is used in the 8088 vectored interrupt scheme. It is a single-byte value that the CPU internally multiplies by four to form a pointer. The pointer points to an interrupt table in segment 0000 hex containing the address of the interrupt service routine. The multiplication of the type number (vector) by four allows the complete physical address (Code Segment and Instruction Pointer, two 16-bit numbers) of the interrupt service routine to be stored in the interrupt table as four consecutive bytes.

The 7863-3 8088 CPU can respond to both internal and external (to the CPU) interrupts. Sources for external interrupts on the 7863-3 can include the STD BUS signals NMIRQ* (nonmaskable interrupt request) and the INTRQ* (interrupt request) and the onboard Serial Communications Controller's interrupt output.

Nonmaskable interrupt requests on the STD BUS NMIRQ* signal are predefined by the 8088 as Type 2 interrupts and no other vector is needed. Figure 5-1 shows how the

CPU processes an NMIRQ*. A Type 2 interrupt has a higher priority than any other external interrupt and since it is non-maskable (the CPU always responds to a Type 2 interrupt at the end of its current instruction cycle), NMIRQ* is generally used to signal the CPU of a "catastrophic" event such as the imminent loss of power or a parity error.

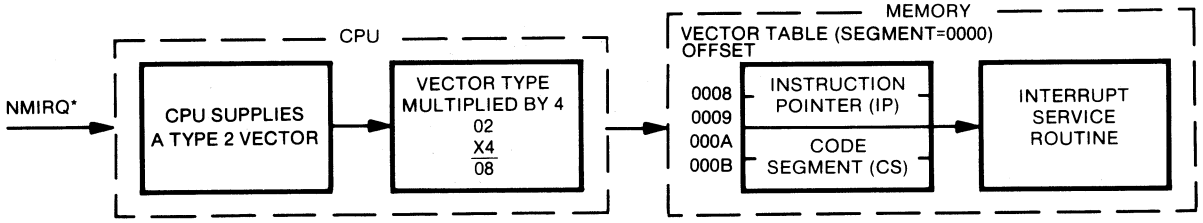


Figure 5-1. NMIRQ* Processing

Interrupt requests on the STD BUS INTRQ* signal and from the onboard SCC are not assigned a type number by the CPU. The interrupting device must provide the type number (or vector) on the CPU's data bus during the interrupt acknowledge cycle. Figure 5-2 shows how the 8088 processes these interrupts. The 7863-3 differentiates between STD BUS INTRQ* and interrupt requests from the onboard SCC; INTRQ* has a higher priority than onboard interrupts. When the 7863-3 responds to an onboard interrupt request, it is assumed that the vector will be generated on-card (by the SCC) and an STD BUS access is not attempted during the interrupt acknowledge cycle.

The 7863-3 responds to an STD BUS INTRQ* by executing an interrupt acknowledge cycle, and reading the type number (vector) supplied by the interrupting device from the STD BUS data bus.

The 7863-3 is capable of generating a vector onboard. INTRQ* is connected (through a buffer) to an unused handshake line on the SCC. The SCC can be programmed to generate an interrupt when the handshake line changes and to provide the required on-card vector when the interrupt is acknowledged. This onboard vector capability allows the use of cards that can generate INTRQ* but not an 8088 vector in a 7863-3 system. The shorting connector should be removed from Jumper W12 if this onboard INTRQ* response is to be used, so that the 7863-3 does not respond to an STD BUS INTRQ* with a bus interrupt acknowledge cycle.

If the shorting connector is removed from Jumper W3 on the 7863-3, the card will not respond to STD BUS INTRQ* at all. Refer to *Interrupt Options* in Section 3 for more information on maskable interrupt response.

The 8087 Math Coprocessor can generate interrupts when certain internal conditions or "exceptions" (such as stack overflow or underflow or indeterminate operations such as divide by zero) are detected. The 8087 interrupt output drives an unused handshake line input on the SCC that can be programmed to provide an on-card interrupt and vector for 8087 interrupts. SCC handshake interrupts are both rising- and falling-edge sensitive rather than level sensitive so an 8087 interrupt (or an INTRQ* "onboard" interrupt as mentioned above) causes two SCC External/Status interrupts -- one when the interrupt occurs and one when the interrupt is cleared. The interrupt service rou-

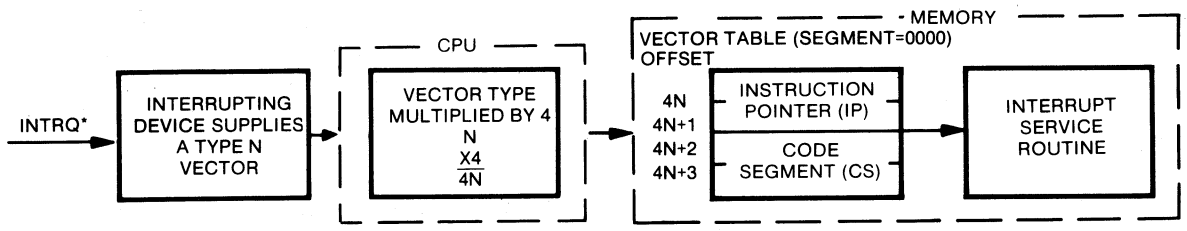


Figure 5-2. INTRQ* Processing

tine must first check the actual state of the handshake line by reading an SCC register to determine whether or not the interrupt needs service.

Memory Addressing

The 7863-3 8088 CPU provides a 20-bit memory address during each memory operation so the 7863-3 can direct address 2^{20} (1,048,576 or 1M) bytes of memory. The two onboard memory sockets can be configured to support various sizes of memory devices and can be mapped at different locations within the one-megabyte memory map. See Figure 3-8 for onboard memory mapping options. Generally, one memory socket is mapped at the high end of memory to support the 8088 power-on restart address (FFFF0 hex) while the other is mapped at the low end to include the 8088 interrupt vector table (00000 to 007FF hex).

When the 7863-3 is addressing off-card memory, the lower 16 address bits are output on address lines A0-A15 on the STD BUS. The upper four bits (A16-A19) are time multiplexed onto the STD data bus (bits D0-D3) at the beginning of each machine cycle. Memory cards used in 8088 STD systems latch these upper address lines on the rising edge of STD BUS control signal MCSYNC*.

MEMEX is defined in the 8088 STD BUS specification as an enable for memory cards that only decode a 16-bit memory address. No provision is made for driving MEMEX on the 7863-3 and only memory cards with the onboard latch required to decode the full 20-bit 8088 addressing should be used with the 7863-3 to avoid redundant addressing of memory.

Several examples of memory mapping in the 7863-3 STD systems are shown in Figure 5-3. Note that the high and low ends of the memory map are usually the onboard memory exclusively available to the 7863-3 and most of the memory map (always at least 792K) is available as common memory.

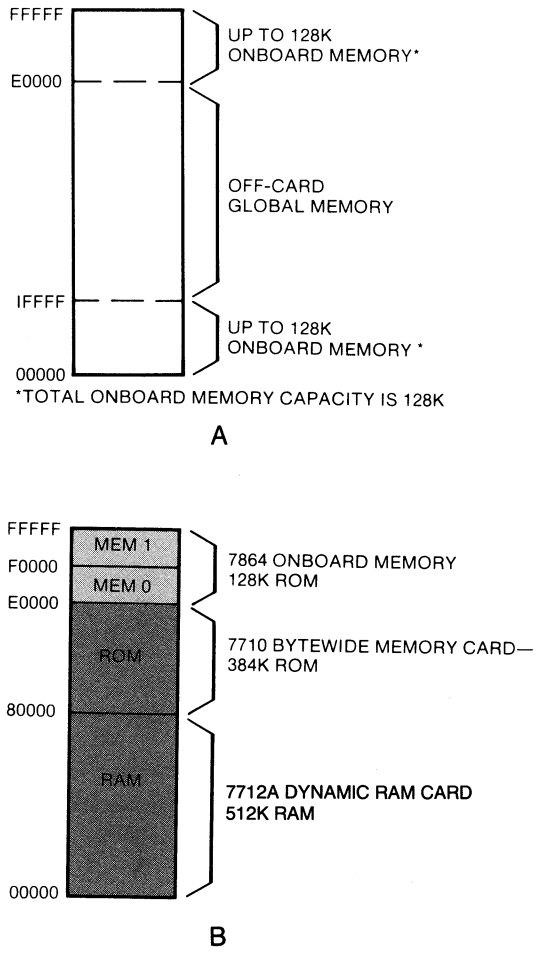


Figure 5-3. Memory Mapping Examples

I/O Addressing

The 7863-3 8088 CPU provides a 16-bit I/O port address during each I/O operation and can directly address 2^{16} (65,536 or 64K) I/O port locations. As with memory, onboard I/O (the SCC) can be read from or written to without having access to the STD BUS. The SCC uses ports BFFC-BFFF hex. See Figure 1-4 for port definitions.

Due to onboard decoding limitations, the upper half of the 64K I/O address field is reserved for onboard I/O even though only ports BFFC-BFFF and port FFFF are actually used. The lower 32K (ports 0000-7FFF hex) is available for common I/O address-

able by the 7863-3. See Figure 1-5 for 7863-3 I/O mapping. Note that ports 8000-BFFB and C000-FFFF hex should never be accessed since the onboard I/O devices will be redundantly addressed in this address range.

Most existing STD BUS I/O cards decode only an 8- or 10-bit I/O address since few system need more than 256 or 1024 ports. The full port address range of these cards (00-FB for 8-bit or 000-3FB for 10-bit) is available. Indirect (16-bit) port addressing should be used when addressing 10-bit I/O cards to allow the card to distinguish between onboard and off-card ports. Eight-bit I/O cards can be directly addressed in Page 00., Eight- and ten-bit I/O cards are not usually used together in the same system since port addresses would have to be selected with great care to avoid redundant addressing ("shadow" ports). Provision is made for grounding the STD BUS IOEXP signal onboard the 7863-3 since many 8-bit address I/O cards use this signal as an enable input. Several examples of I/O port configurations in 7863-3 systems are shown in Figure 5-4.

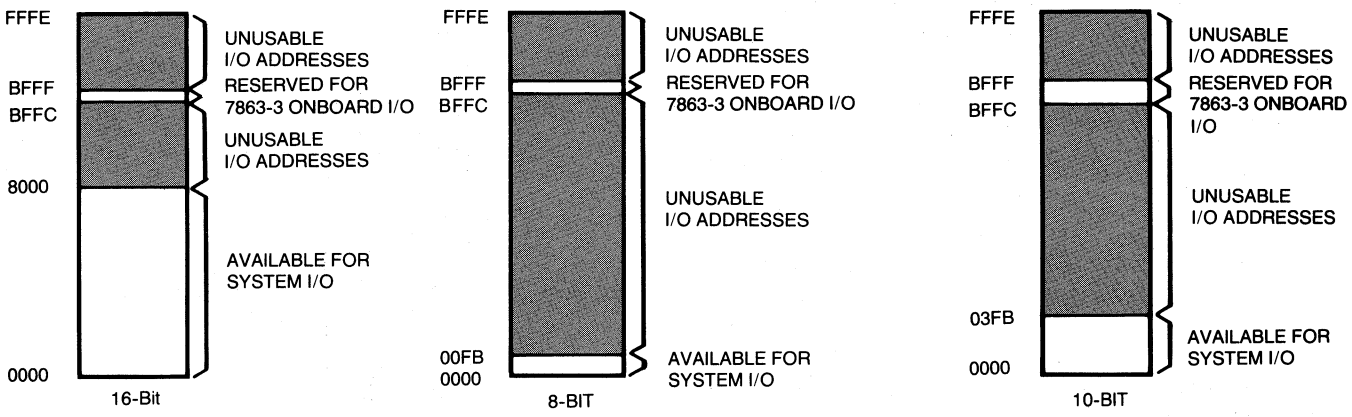


Figure 5-4. I/O Port Configuration Examples (8-, 10-, and 16-Bit)

INTRODUCTION

The Serial Communications Controller (SCC) has two completely independent full-duplex communications channels, each with its own baud rate generator. Channel A drives the RS-232-C interface and channel B drives the RS-422-A interface. The SCC is capable of transmitting and receiving data in various formats: asynchronous, bit synchronous, and byte synchronous. Examples of each are shown in Figure 6-1.

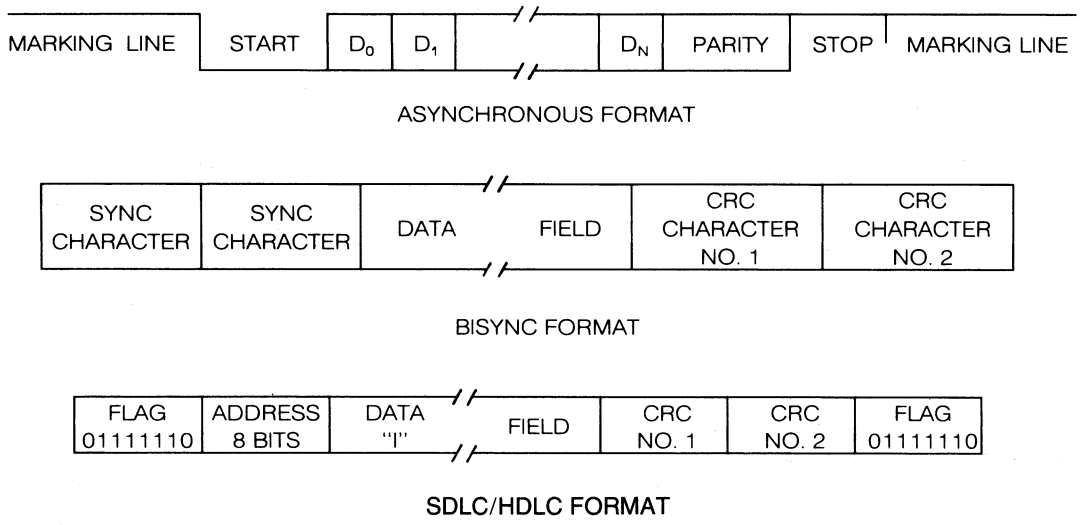


Figure 6-1. Examples of Various Formats

The RS-232-C interface should be configured as DTE or DCE as required by the application (see Section 2, *Installation*, for details). The maximum baud rate allowed in the RS-232-C specification is 19.2K baud. However, the SCC can be operated at much higher baud rates both asynchronously and synchronously. A discussion of RS-232-C handshaking is contained in the following paragraphs.

The RS-422-A interface provides a minimum number of wires for adequate synchronous data transmission. Data, clock, and ground signals are provided. All handshaking is accomplished in software. The pinout of the RS-422-A connector was selected to eliminate the need for DTE/DCE configuration jumpers by making it doubly symmetrical, as described in Section 2, *Installation.*

The RS-422-A is used in serial data communications applications that require long cable lengths, high noise immunity, or high data rates. Using a 24 AWG twisted-pair line, the 7863 RS-422-A port can transmit or receive data over a maximum distance of 4,000 feet (1.2 kilometers) at synchronous data rates as high as 90K baud.

The maximum permissible length of cable separating an RS-422-A transmitter and receiver is a function of the data signaling rate and is influenced by the allowable signal distortion, the amount of longitudinally coupled noise, the difference in ground potential introduced between the transmitter and receiver grounds, and cable balance.

Increasing the cable length between transmitter and receiver increases exposure to common mode noise, signal distortion, and the effects of cable imbalance.

The curve of cable length versus data signaling rate given in Figure 6-2, from the EIA RS-422-A standard, may be used as a conservative guide. This curve is based on empirical data using a 24 AWG, copper conductor, twisted-pair telephone cable with a shunt capacitance of 52.5 pF/meter (16 pF/foot) terminated in a 100-ohm resistive load. The cable length restriction shown by the curve

is based on assumed load signal quality requirements of:

a. Signal rise and fall time equal to or less than one-half unit interval at the applicable modulation rate.

b. A maximum voltage loss between generator and load of 6 db (voltage).

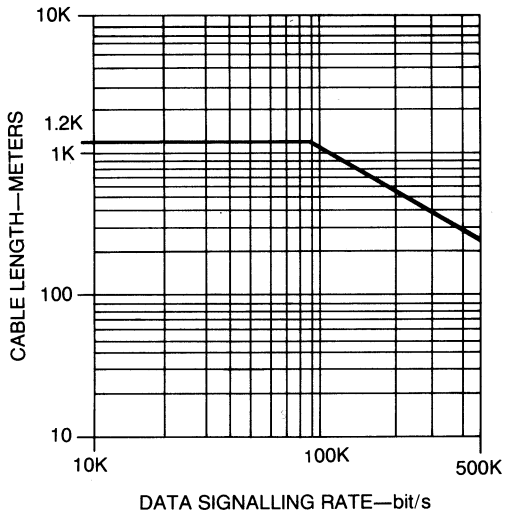


Figure 6-2. Data Signaling Rate vs Cable Length for Balanced Interface

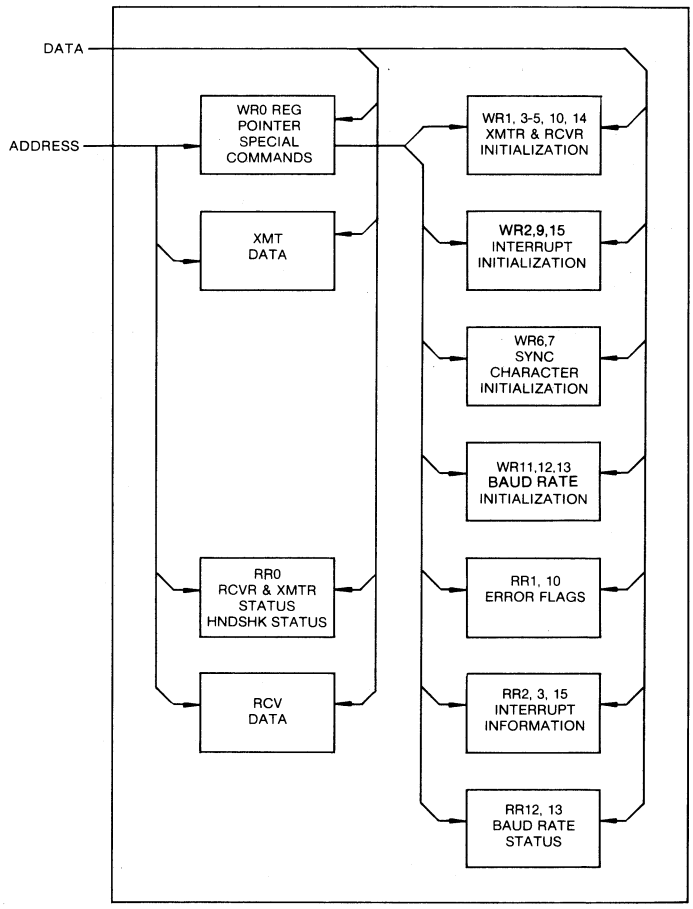


Figure 6-3. SCC Register Structure

INTERNAL STRUCTURE

Operation of the serial ports depends upon the contents of the SCC internal control (write) and status (read) registers which are accessed through I/O ports. Due to SCC cycle time limitations, consecutive data transfer instructions between SCC registers and the 8088 CPU internal registers must always be separated by at least one other instruction in the program listing. A NOP (No Operation) should be used if no other instruction is actually needed. Figure 6-3 shows the structure of the SCC registers.

Each SCC channel contains 16 write registers: 10 for control, 2 for sync character generation, and 2 for baud rate generation. The remaining 2 registers are shared by both channels; one is used as the interrupt vector and one as the master interrupt control. Each channel also contains eight read registers: four registers indicate status, two are used by the baud rate generator, one for the interrupt vector, and one for the receive buffer. Channel A has an additional register for reading the interrupt pending bits of both channels. Figures 6-4 and 6-5 show the basic functions of the registers. Some of the functions of the SCC have not been tested by Pro-Log on the 7863. Further information concerning features of the SCC may be obtained by acquiring a copy of the Zilog Z8530 SCC technical manuals.

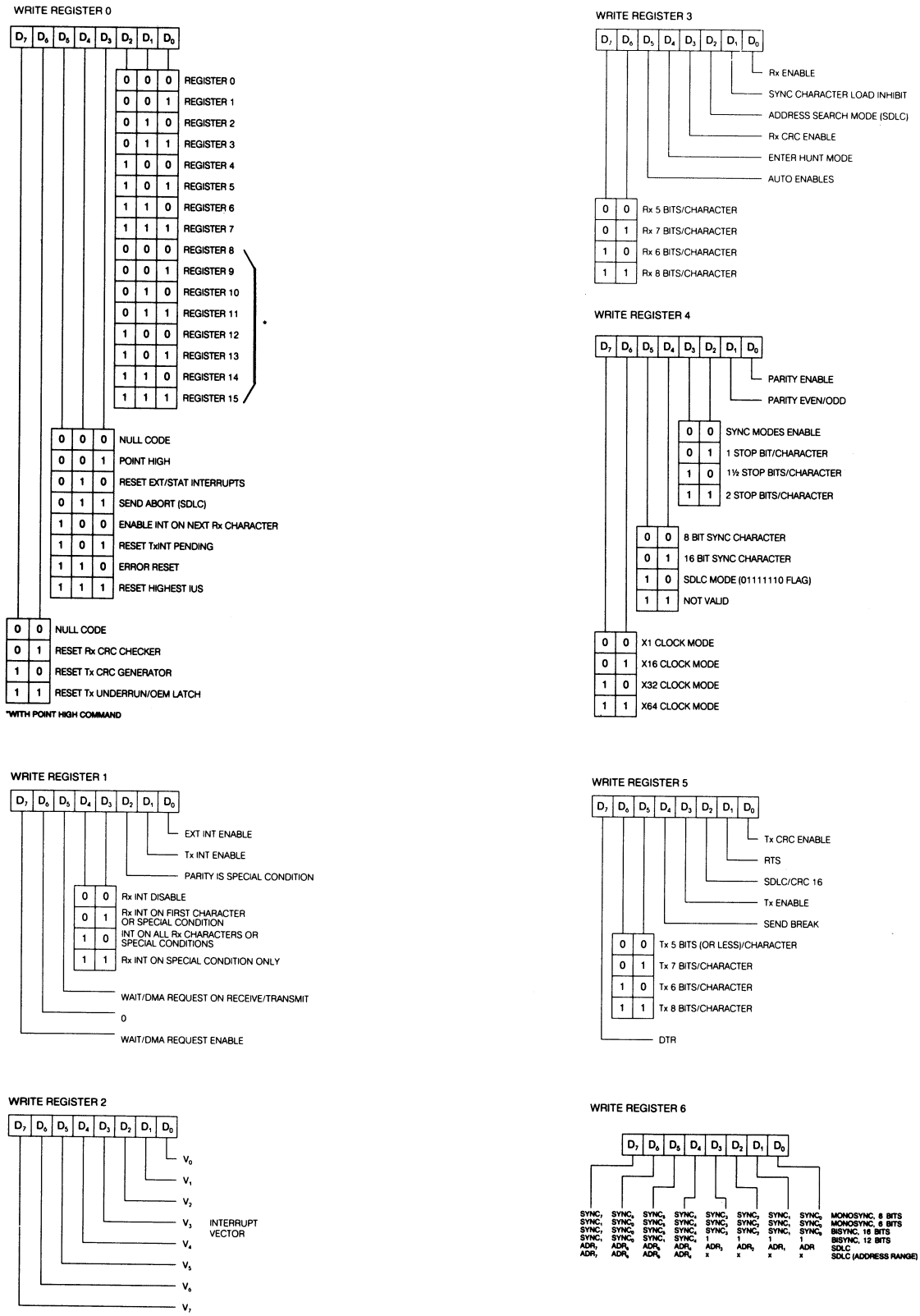

WRITE REGISTER 0
D7 D6 D5 D4 D3 D2 D1 D0
0 0 0 REGISTER 0
0 0 1 REGISTER 1
0 1 0 REGISTER 2
0 1 1 REGISTER 3
1 0 0 REGISTER 4
1 0 1 REGISTER 5
1 1 0 REGISTER 6
1 1 1 REGISTER 7
0 0 0 REGISTER 8
0 0 1 REGISTER 9
0 1 0 REGISTER 10
0 1 1 REGISTER 11
1 0 0 REGISTER 12
1 0 1 REGISTER 13
1 1 0 REGISTER 14
1 1 1 REGISTER 15
0 0 0 NULL CODE
0 0 1 POINT HIGH
0 1 0 RESET EXT/STAT INTERRUPTS
0 1 1 SEND ABORT (SDLC)
1 0 0 ENABLE INT ON NEXT Rx CHARACTER
1 0 1 RESET TxINT PENDING
1 1 0 ERROR RESET
1 1 1 RESET HIGHEST IUS
0 0 NULL CODE
0 1 RESET Rx CRC CHECKER
1 0 RESET Tx CRC GENERATOR
1 1 RESET Tx UNDERRUN/OEM LATCH
*WITH POINT HIGH COMMAND
WRITE REGISTER 3
D7 D6 D5 D4 D3 D2 D1 D0
Rx ENABLE
SYNC CHARACTER LOAD INHIBIT
ADDRESS SEARCH MODE (SDLC)
Rx CRC ENABLE
ENTER HUNT MODE
AUTO ENABLES
0 0 Rx 5 BITS/CHARACTER
0 1 Rx 7 BITS/CHARACTER
1 0 Rx 6 BITS/CHARACTER
1 1 Rx 8 BITS/CHARACTER
WRITE REGISTER 4
D7 D6 D5 D4 D3 D2 D1 D0
PARITY ENABLE
PARITY EVEN/ODD
0 0 SYNC MODES ENABLE
0 1 1 STOP BIT/CHARACTER
1 0 1½ STOP BITS/CHARACTER
1 1 2 STOP BITS/CHARACTER
0 0 8 BIT SYNC CHARACTER
0 1 16 BIT SYNC CHARACTER
1 0 SDLC MODE (01111110 FLAG)
1 1 NOT VALID
0 0 X1 CLOCK MODE
0 1 X16 CLOCK MODE
1 0 X32 CLOCK MODE
1 1 X64 CLOCK MODE
WRITE REGISTER 1
D7 D6 D5 D4 D3 D2 D1 D0
EXT INT ENABLE
Tx INT ENABLE
PARITY IS SPECIAL CONDITION
0 0 Rx INT DISABLE
0 1 Rx INT ON FIRST CHARACTER OR SPECIAL CONDITION
1 0 INT ON ALL Rx CHARACTERS OR SPECIAL CONDITIONS
1 1 Rx INT ON SPECIAL CONDITION ONLY
WAIT/DMA REQUEST ON RECEIVE/TRANSMIT
0
WAIT/DMA REQUEST ENABLE
WRITE REGISTER 5
D7 D6 D5 D4 D3 D2 D1 D0
Tx CRC ENABLE
RTS
SDLC/CRC 16
Tx ENABLE
SEND BREAK
0 0 Tx 5 BITS (OR LESS)/CHARACTER
0 1 Tx 7 BITS/CHARACTER
1 0 Tx 6 BITS/CHARACTER
1 1 Tx 8 BITS/CHARACTER
DTR
WRITE REGISTER 2
D7 D6 D5 D4 D3 D2 D1 D0
V0
V1
V2
V3 INTERRUPT VECTOR
V4
V5
V6
V7
WRITE REGISTER 6
D7 D6 D5 D4 D3 D2 D1 D0
MONOSYNC, 8 BITS
MONOSYNC, 6 BITS
BISYNC, 16 BITS
BISYNC, 12 BITS
SDLC
SDLC (ADDRESS RANGE)

Figure 6-4. Write Register Bit Maps

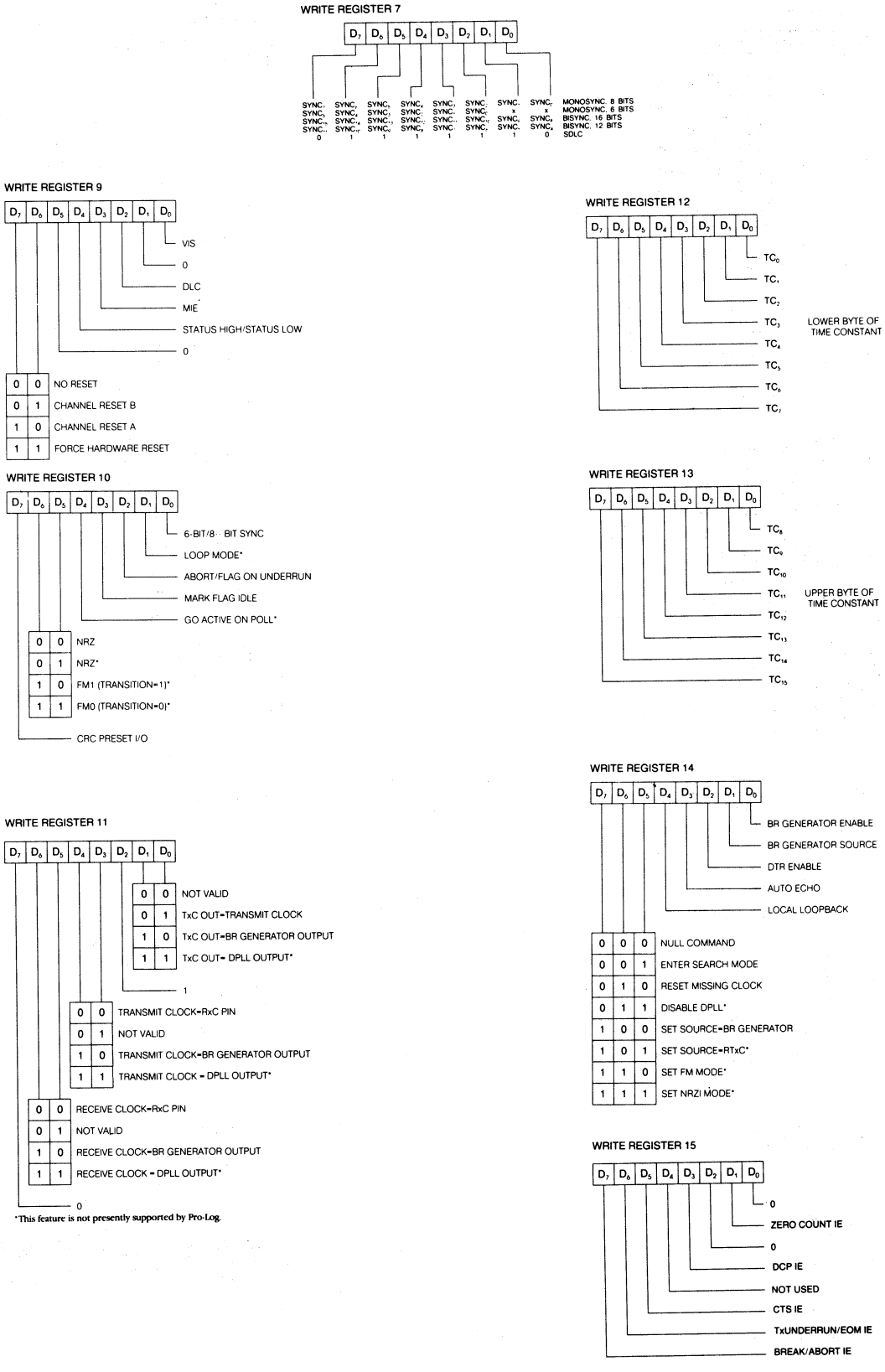
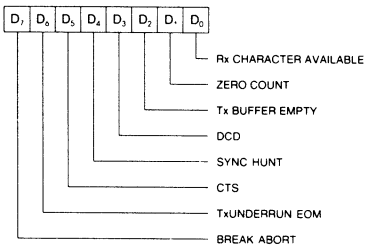
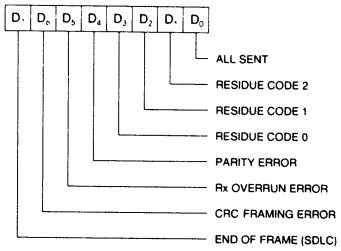


Figure 6-4. (Continued)

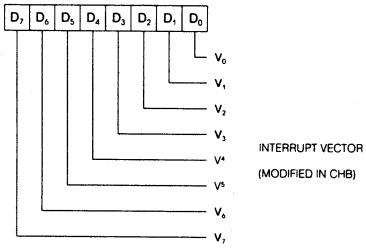
READ REGISTER 0



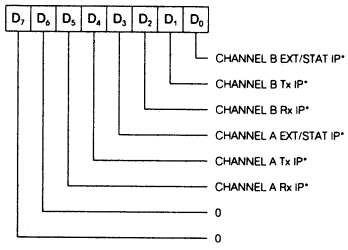
READ REGISTER 1



READ REGISTER 2



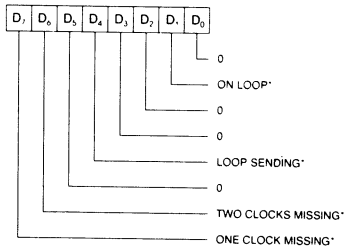
READ REGISTER 3



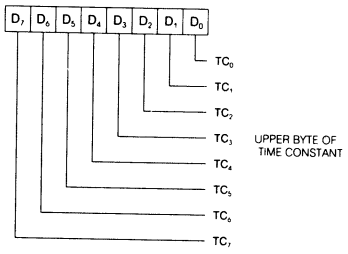
*ALWAYS 0 IN B CHANNEL

*This feature is not presently supported by Pro-Log on the 7863.

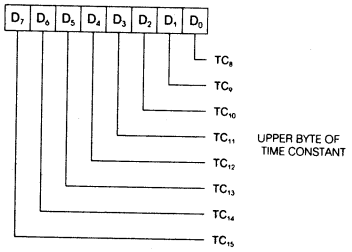
READ REGISTER 10



READ REGISTER 12



READ REGISTER 13



READ REGISTER 15

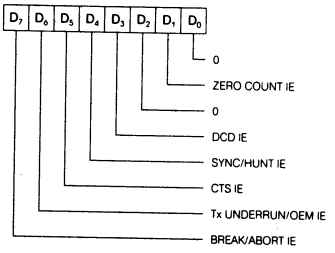


Figure 6-5. Read Register Bit Maps

SCC RECEIVE AND TRANSMIT REGISTERS

The receive (RCV) data register is directly addressable and holds the data that is received over the serial interface. It holds up to three bytes in addition to the byte currently being received. The error flags in the read register are also four levels deep so the appropriate status information is available with each byte. RR1 should be read before the RCV data register as the stacks for both data and status are incremented when the data is read. If enabled, an interrupt occurs each time a character reaches the top of the stack. If there are no characters pending in the stack, a received character goes directly to the top of the stack.

If four complete bytes have been received since the last time the processor read a byte out of the RCV data register, the third byte will be overwritten. The fourth character, in the shift register, is loaded into the bottom of the stack of received characters as soon as it is completely received. If the Status Affects Vector mode is selected, an interrupt is generated with a special vector indicating an error. This interrupt occurs when the character that overwrote the previous one reaches the top of the stack. An error flag is set in the status information associated with the fourth byte, indicating that it overwrote the previous byte.

If an error occurs in the Interrupt on First Received Character mode (interrupts can occur on the first character or every character), the data and status stacks are frozen, even if read, and prevent access to further data until an error reset is performed. Error reset is explained in the WR0 description.

The transmit (XMT) data register is directly addressable and holds the data to be transmitted. In addition to the XMT data register, there is a 20-bit shift register for the byte currently being transmitted. This allows the processor to give the SCC one byte while it is transmitting another, thereby eliminating delays between characters.

An interrupt and/or flag indicates that the XMT data register is empty and is able to accept another character. The All Sent flag in RR1 indicates that actual transmission has been completed. The All Sent condition does not generate an interrupt.

SCC CONTROL REGISTERS

All control registers require two bytes to be programmed except for WR0. The first byte contains the register select code in the least significant three bits and, if the code is for WR0, the remainder of the byte is the control word. For the other registers, the control byte is the next word. The first byte written into each channel after a reset (either with a channel reset command or hardware system reset) is directed to WR0. Figure 6-6 lists the functions assigned to each control register. Detailed descriptions of these registers are given on the following pages. All references to pin states are as seen at the SCC. The state programmed into a bit is the state that appears at the pin. The RS-232-C interface (channel A) inverts the signal. Pro-Log presently does not support some features of the SCC. These are marked with an asterisk (*) in the register diagram.

WR0	CRC initialize, initialization commands for the various modes, register pointers
WR1	Transmit/Receive interrupt and data transfer mode definition
WR2	Interrupt vector (accessed through either channel of each SCC)
WR3	Receive parameters and control
WR4	Transmit/Receive miscellaneous parameters and modes
WR5	Transmit parameters and controls
WR6	Sync characters or SDLC address field
WR7	Sync character of SDLC flag
WR8	Transmit buffer
WR9	Master interrupt control and reset (accessed through either channel of each SCC)
WR10	Miscellaneous transmitter/receiver control bits
WR11	Clock mode control
WR12	Lower byte of baud rate generator time constant
WR13	Upper byte of baud rate generator time constant
WR14	Miscellaneous control bits
WR15	External/Status interrupt control

Figure 6-6. SCC Control Register Functions

Write Register 0

This register controls various reset conditions, CRC initializations, and contains register pointers, allowing access to the remaining fourteen registers.

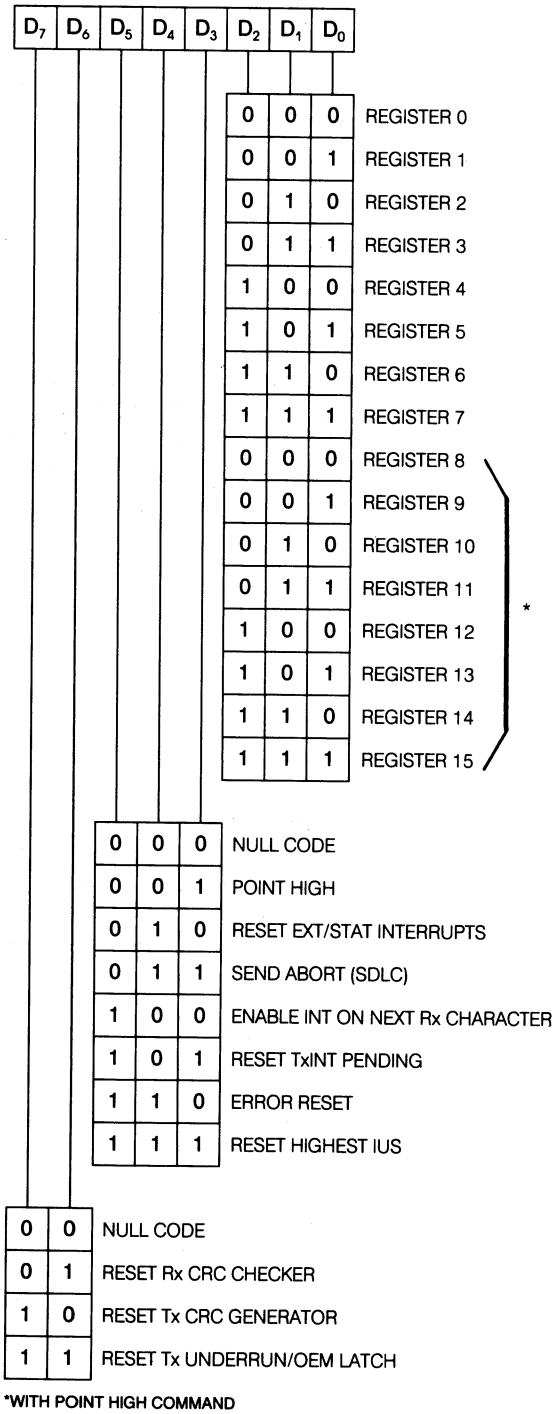

D7 D6 D5 D4 D3 D2 D1 D0
0 0 0 REGISTER 0
0 0 1 REGISTER 1
0 1 0 REGISTER 2
0 1 1 REGISTER 3
1 0 0 REGISTER 4
1 0 1 REGISTER 5
1 1 0 REGISTER 6
1 1 1 REGISTER 7
0 0 0 REGISTER 8
0 0 1 REGISTER 9
0 1 0 REGISTER 10
0 1 1 REGISTER 11
1 0 0 REGISTER 12
1 0 1 REGISTER 13
1 1 0 REGISTER 14
1 1 1 REGISTER 15
*
0 0 0 NULL CODE
0 0 1 POINT HIGH
0 1 0 RESET EXT/STAT INTERRUPTS
0 1 1 SEND ABORT (SDLC)
1 0 0 ENABLE INT ON NEXT Rx CHARACTER
1 0 1 RESET TxINT PENDING
1 1 0 ERROR RESET
1 1 1 RESET HIGHEST IUS
0 0 NULL CODE
0 1 RESET Rx CRC CHECKER
1 0 RESET Tx CRC GENERATOR
1 1 RESET Tx UNDERRUN/OEM LATCH
*WITH POINT HIGH COMMAND

Figure 6-7. Write Register 0 Bit Map

D7-D6 Reset Bits

These bits specify three reset modes: reset receive CRC check, reset transmit CRC generator, and reset sending transmit underrun/end of message latch.

D5-D3 Command Bits

Command 0 — This is a "no operation" command that can be issued while setting the register pointer when no other effect is desired.

Command 1 — Write Registers WR8 through WR15 are accessed with this command.

Command 2 — After the status bits of RR0 have been latched by a change of one of the external/status bits, this command reenables the latch of RR0. If an external/status interrupt occurs, command 2 is issued to reenable RR0, which allows external/status interrupts to occur. The latching allows capture of a short pulse on the input pins and holds the pulse until the CPU can read it.

Command 3 — A sequence of 8 to 13 ones is generated. It is used only with the SDLC mode.

Command 4 — If the Interrupt Only on First Received Character mode is selected (D4, D3, WR1), it must be reactivated after each complete message is received.

Command 5 — The transmitter interrupts when the transmit buffer becomes empty, if transmit interrupt was enabled at D1 of WR1. This command resets the transmitter interrupt pending until after the next character has been loaded into the transmitter. This is used during periods when no characters are to be sent.

Command 6 Parity and overrun errors are latched in RR1 until reset by this command.

Command 7 This command resets the highest priority Interrupt Under Service (IUS) bit and allows lower priority conditions to request an interrupt. This should be the last operation in an interrupt service routine.

D2-D0 Register Pointers

These bits indicate to which register the next write command is written. The first byte written into each channel after a reset (either with a channel reset command or hardware system reset) is directed to WR0 and the byte following a read or write to any register.

Write Register 1

This register controls interrupt conditions.

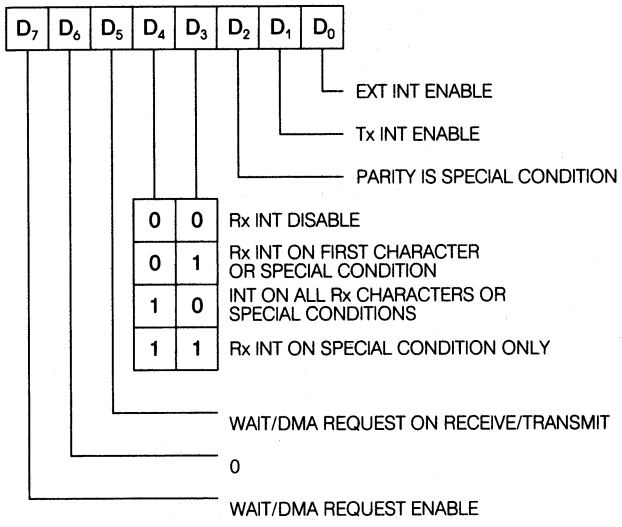

D7 D6 D5 D4 D3 D2 D1 D0
EXT INT ENABLE
Tx INT ENABLE
PARITY IS SPECIAL CONDITION
0 0 Rx INT DISABLE
0 1 Rx INT ON FIRST CHARACTER OR SPECIAL CONDITION
1 0 INT ON ALL Rx CHARACTERS OR SPECIAL CONDITIONS
1 1 Rx INT ON SPECIAL CONDITION ONLY
WAIT/DMA REQUEST ON RECEIVE/TRANSMIT
0
WAIT/DMA REQUEST ENABLE

Figure 6-8. Write Register 1 Bit Map

D7 Wait Request Enable

This bit enables the wait request function of the SCC. If bit 7 is set to 1, the 8088 CPU on the 7863 will be held in a wait state if it attempts to transfer data before the SCC is ready.

D6 The 7863 requires that this bit be set to zero which defines the W/REQ* pins on the SCC (pins 10 and 30) as open drain, active low.

D5 Wait Request on Receive/Transmit Select

This bit determines whether the wait request function enabled by bit 7 operates in the receive or transmit mode. When set to 1, this bit allows the wait request function to follow the state of the receive buffer, i.e.,

the CPU is held in a wait state if it attempts to read data when the selected channel's receive buffer is empty. If this bit is set to 0, the wait request function follows the state of the transmit buffer (the CPU is held in a wait state if it attempts to write data to the selected channel's transmit buffer when it is full).

D4-D3 Receiver Interrupt Mode

These bits specify the character available interrupt conditions. If the Vector Includes Status is enabled, the specific vector generated depends upon whether or not an error occurs. The special receive condition interrupt occurs if there is a receiver error, otherwise the receive character available interrupt occurs.

D2 Parity is special condition

If D2 is set, a parity error results in a special condition and modifies the interrupt vector. If parity is disabled (D0, WR4), this bit is ignored.

D1 Transmitter Interrupt Enable

This bit allows interrupts whenever the transmitter buffer becomes empty. This interrupt generates the Transmit Buffer Empty vector of RR2 if the Vector Includes Status (VIS) bit is set (D0, WR9).

D0 External Interrupt Enable

This bit allows interrupts to occur as a result of transitions on the DCD*, CTS*, or SYNC* (external interrupt) lines, or as a result of a break condition (read at RR0, or the beginning of

Write Register 2

This is the interrupt vector register. Only one vector register exists in each SCC. It may be programmed with a vector (interrupt type number) for the 8088. Some of the bits may be programmed to change according to SCC status.

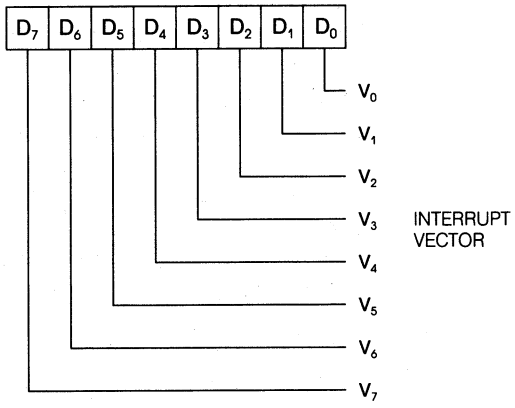


Figure 6-9. Write Register 2 Bit Map

Write Register 3

This register contains receive control parameters, the number of bits per character, and the receive enable.

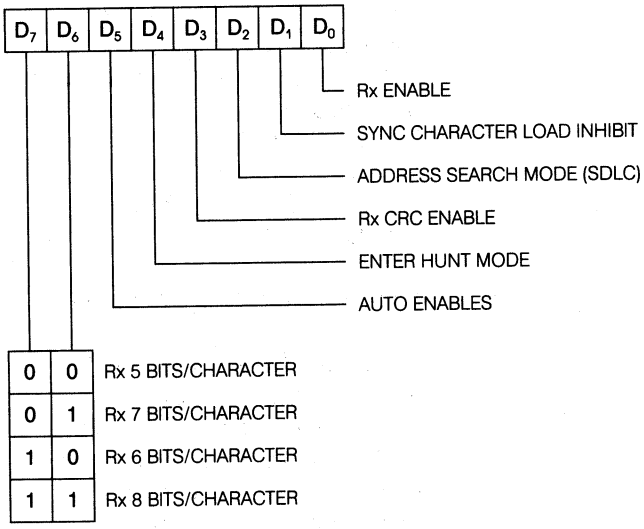

D7 D6 D5 D4 D3 D2 D1 D0
Rx ENABLE
SYNC CHARACTER LOAD INHIBIT
ADDRESS SEARCH MODE (SDLC)
Rx CRC ENABLE
ENTER HUNT MODE
AUTO ENABLES
0 0 Rx 5 BITS/CHARACTER
0 1 Rx 7 BITS/CHARACTER
1 0 Rx 6 BITS/CHARACTER
1 1 Rx 8 BITS/CHARACTER

Figure 6-10. Write Register 3 Bit Map

sending CRC or sync characters. These interrupts show in the external/status change vector interrupts modify the vector of RR2 if the VIS bit is programmed (D0, WR9).

D7-D6 Receiver Bits/Character

These bits determine the number of serial receive bits that are assembled to form a character. An additional parity bit is received if parity is enabled at D0 of WR4.

D5 Auto Enables

If this mode is selected, the DCD* and CTS* lines for the channel are receiver and transmitter enables, respectively. If the mode is not selected, the DCD* and CTS* are inputs to bits in RR0. Transitions on these lines can cause an interrupt if external interrupts are enabled at D0 of WR1.

D4 Enter Hunt Mode

If character synchronization is lost or, if in the SDLC mode, the incoming message is not needed, the Hunt mode may be re-entered by setting this bit.

D3 Receiver CRC Enable

When this bit is set, CRC calculation begins at the start of the last character transferred from the receive register to the buffer stack. This occurs regardless of the number of characters in the stack.

Write Register 4

WR4 contains transmit and receive parameters and modes, clock rate, number of sync characters, stop bits, and parity.

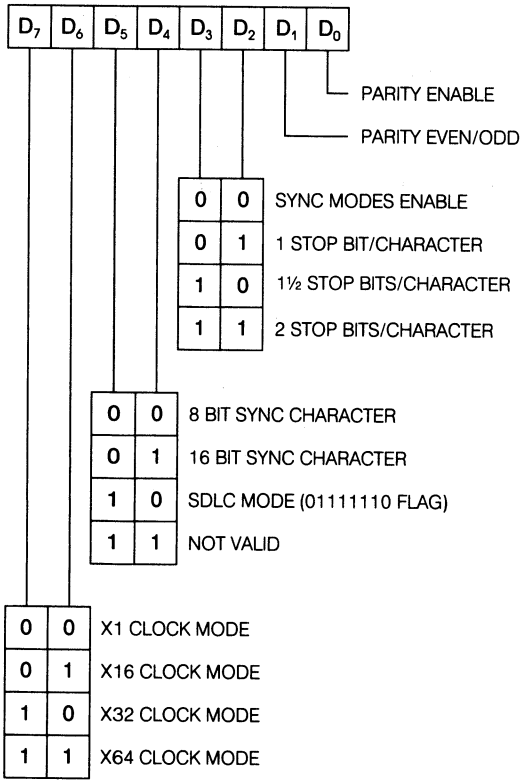

D7 D6 D5 D4 D3 D2 D1 D0
PARITY ENABLE
PARITY EVEN/ODD
0 0 SYNC MODES ENABLE
0 1 1 STOP BIT/CHARACTER
1 0 1½ STOP BITS/CHARACTER
1 1 2 STOP BITS/CHARACTER
0 0 8 BIT SYNC CHARACTER
0 1 16 BIT SYNC CHARACTER
1 0 SDLC MODE (01111110 FLAG)
1 1 NOT VALID
0 0 X1 CLOCK MODE
0 1 X16 CLOCK MODE
1 0 X32 CLOCK MODE
1 1 X64 CLOCK MODE

Figure 6-11. Write Register 4 Bit Map

D2 Address Search Mode

When this bit is set and the SDLC mode is selected, no interrupts are generated unless an address match (programmed or global) occurs.

D1 Sync Character Load Inhibit

When this bit is set, sync characters preceding a message are not loaded into the receive buffers. The CRC calculation is not terminated if no sync characters are generated.

D0 Receiver Enable

This bit enables the receiver and is normally left set throughout operation. If the receiver is disabled while receiving a character, the character is lost and no interrupt occurs.

D7-D6 Clock Rate

The clock rate specifies the multiplier between clock and data rates. The same multiplier is used for both the receiver and transmitter. In all modes, the system clock must be at least five times the data rate. If the X1 clock rate is selected, bit synchronization must be accomplished externally.

D5-D4 Sync Modes

These bits select character synchronization options.

D3-D2 Stop Bits

These bits determine the number of stop bits added to each character sent. The

receiver always checks for one stop bit.

D1 Parity Even/Odd

If parity is specified by bit D0, this bit determines whether the parity is sent or checked as even (D1=1) or odd (D1=0).

D0 Parity

When this bit is set, an additional bit position (in addition to those specified in the bits/character control of WR3 or 4) is added to transmitted data and is expected in receive data.

Write Register 5

WR5 contains transmit parameters and controls, number of transmit bits per character, and the transmit CRC enable.

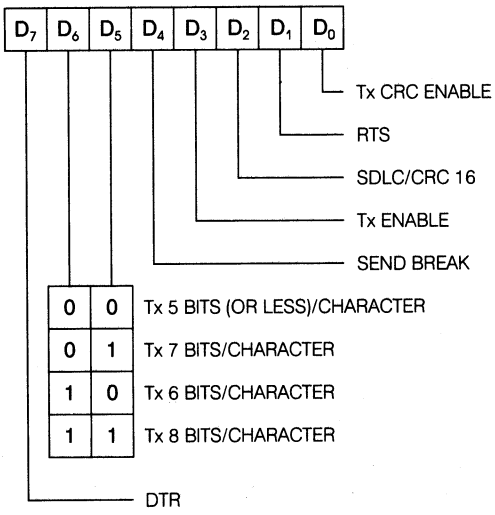


Figure 6-12. Write Register 5 Bit Map

D7 Data Terminal

This is the control bit for the DTR* output pin for the channel. When set, DTR* is active low. When reset, DTR* is inactive high. On channel B, DTR* enables the RS-422 drivers when high and disables them when low.

D6-D5 Transmit Bits/Character

These bits determine the number of bits sent from each byte transferred to the transmit buffer. An additional parity bit is sent if parity is enabled at D0 of WR4. Bits are assumed to be right justified. If the 5 bits/character mode is selected, the unused bits of each data byte must be zero.

D4 Send Break

When set, this bit forces the TxD pin for the channel to spacing low, regardless of any data being transmitted. When reset, the TxD pin for the channel is released.

Write Register 6

WR6 contains the first 8 bits of a BISYNC sequence. It must contain the check address (if used) for SDLC mode and the sync character in the 8-bit sync mode.

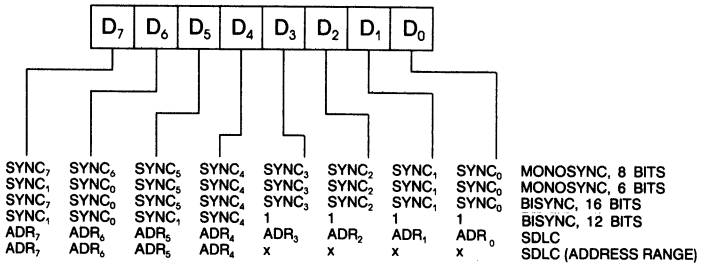

D7 D6 D5 D4 D3 D2 D1 D0
SYNC7 SYNC6 SYNC5 SYNC4 SYNC3 SYNC2 SYNC1 SYNC0 MONOSYNC, 8 BITS
SYNC1 SYNC0 SYNC5 SYNC4 SYNC3 SYNC2 SYNC1 SYNC0 MONOSYNC, 6 BITS
SYNC7 SYNC0 SYNC5 SYNC4 SYNC3 SYNC2 SYNC1 SYNC0 BISYNC, 16 BITS
SYNC1 SYNC0 SYNC1 SYNC4 1 1 1 1 BISYNC, 12 BITS
ADR7 ADR6 ADR5 ADR4 ADR3 ADR2 ADR1 ADR0 SDLC
ADR7 ADR6 ADR5 ADR4 x x x x SDLC (ADDRESS RANGE)

Figure 6-13. Write Register 6 Bit Map

Write Register 7

WR7 contains an 8-bit sync character of a 16-bit synchronization sequence, or the SDLC flag 01111110.

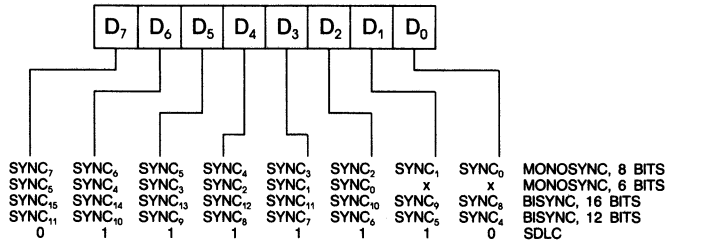

D7 D6 D5 D4 D3 D2 D1 D0
SYNC7 SYNC6 SYNC5 SYNC4 SYNC3 SYNC2 SYNC1 SYNC0 MONOSYNC, 8 BITS
SYNC5 SYNC4 SYNC3 SYNC2 SYNC1 SYNC0 x x MONOSYNC, 6 BITS
SYNC15 SYNC14 SYNC13 SYNC12 SYNC11 SYNC10 SYNC9 SYNC8 BISYNC, 16 BITS
SYNC11 SYNC10 SYNC9 SYNC8 SYNC7 SYNC6 SYNC5 SYNC4 BISYNC, 12 BITS
0 1 1 1 1 1 1 0 SDLC

Figure 6-14. Write Register 7 Bit Map.

Write Register 8

WR8 is the transmit buffer.

D3 Transmit Enable

This bit enables the transmitter. Data is not transmitted and the transmit data pin remains high until the transmitter is enabled. If the transmitter is disabled while sending, the character being sent will be completed. An interrupt still occurs when the XMT Data register becomes empty. Only the character being sent is completed. If one is pending in the XMT Data register, it remains pending. CRC characters are not sent if the transmitter is disabled during CRC transmission.

D2 SDLC/CRC

This bit selects the CRC code used by the transmitter and receiver. When the bit is set, the CRC-16 polynomial $X^{16} + X^{15} + X^2 + 1$ is used and the CRC registers are reset to 0. When D2 is reset, the SDLC polynomial $X^{16} + X^{12} + X^5 + 1$ is used.

D1 Request to Send

This is the control bit for the RTS* output pin for the channel. When D1 is set, the RTS* line goes active low. When D1 is reset, the RTS* pin is inactive high only after the transmitter is empty.

D0 Transmit CRC Enable

If this bit is set at the time of loading the character from the transmit buffer to the transmit shift register, the CRC is calculated on the character. The CRC is not automatically sent when the transmitter is empty unless this bit is set.

Write Register 9

WR9 contains the master interrupt control and reset. Both channels of each SCC share WR9 and it can be accessed through either channel. The interrupt control bits can be programmed at the same time as the reset command.

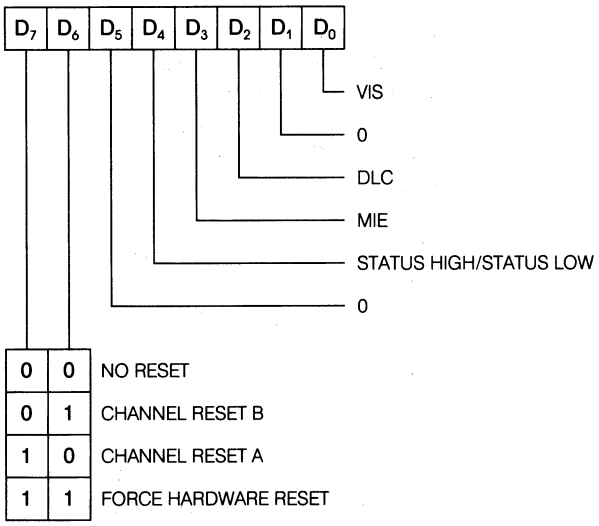


Figure 6-15. Write Register 9 Bit Map

D7-D6 Reset Bits

Setting either D7 or D6 disables the receiver and transmitter, forces the handshake signals high, resets the interrupt pending (IP) and interrupt under service (IUS) bits, and disables all interrupts in the corresponding channel. Four additional PCLK cycles must be allowed before additional command or control bytes are written to each SCC.

NO RESET: This is a "no operation" command that is used when WR9 is written to but no resets are implemented.

CHANNEL RESET B: Channel B is reset

CHANNEL RESET A: Channel A is reset

FORCED HARDWARE RESET: This command has the same effect as a hardware reset except the MIE, status high/status low, and DLC bits are reset.

D5 This bit is not used and must be reset.

D4 Status High/Status Low

When D4 is set, bits V6, V5, and V4 of the interrupt vector will be modified. When D4 is reset, bits V3, V2, and V1 will be modified (see Figure 6-26).

D3 Master Interrupt Enable

SCC interrupts are enabled when this bit is set. When D3 is reset, SCC interrupts are disabled.

D2 This bit is not used on the 7863.

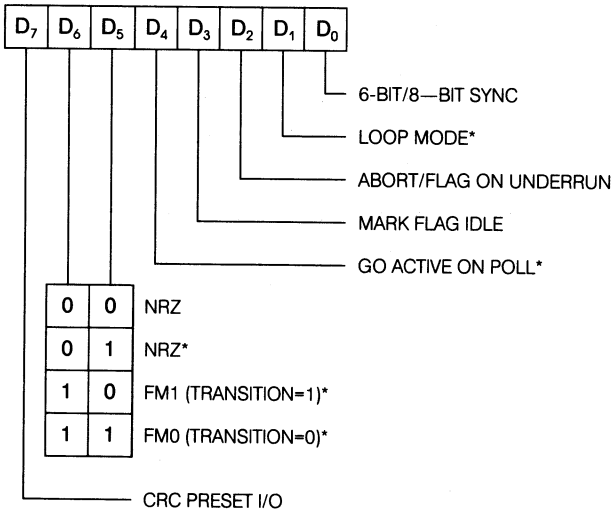
D1 This bit is not used on the 7863 and must be reset.

D0 Vector Includes Status

If this bit is set, the vector returned by the SCC in response to an interrupt acknowledge cycle is variable, according to Figure 6-26. If D0 is reset, the vector returned is that programmed in WR2.

Write Register 10

WR10 contains miscellaneous transmitter and receiver control bits, NRZI, NRZ, and FM encoding options, and the CRC reset.



*This feature is not presently supported by Pro-Log.

Figure 6-16. Write Register 10 Bit Map

D7 CRC Preset I/O

The initialized condition of the received CRC checker and the transmit CRC generator are selected by this bit. If D7 is set, the CRC generator and checker are preset to 1. If D7 is reset, the CRC generator and checker are preset to 0. Either option can be used with either CRC polynomial. This bit is ignored in asynchronous modes.

D6-D5 Data Encoding

These bits control the clock encoding method used for both the transmitter and receiver of the selected SCC channel. Only NRZ clock encoding (D6D5=00) is presently supported by Pro-Log.

D4 Go Active on Poll

This is used for SDLC loop mode operation which is not presently supported by Pro-Log.

D3 Mark/Flag Idle

This bit affects SDLC idle operation. With D3 set, the transmitter sends continuous ones after the closing flag of a frame. If D3 is reset, continuous flags are sent during idle operation.

D2 Abort/Flag on Underrun

In SDLC operation, if D2 is set, the SCC sends an abort and a flag instead of a CRC on a transmit underrun. If D2 is reset, a CRC is sent on a transmit underrun. If both D3 and D2 are set, all ones are transmitted after the transmit underrun. This bit should be set after the first byte of data is sent to the SCC and reset immediately after the last byte of data, so the SDLC frame ends with a CRC and a flag.

D1 Loop Mode

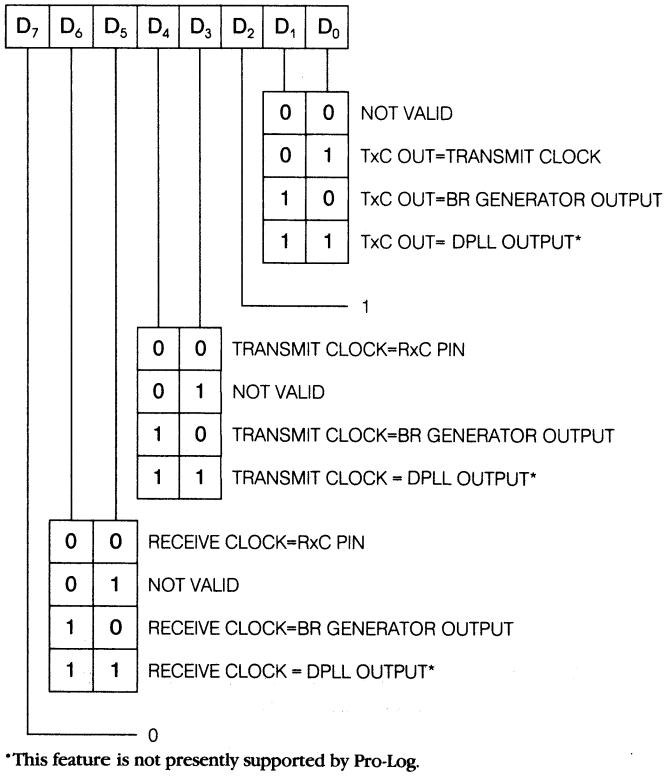
This bit is used for SDLC loop mode operation which is not presently supported by Pro-Log.

D0 6-Bit/8-Bit Sync

This bit selects a synchronous mode. In monosync mode, if D0 is set, the receiver and transmitter sync characters are six bits long instead of eight bits. In BISYNC mode, if D0 is set, the transmitter sync character remains 16 bits long. This bit is ignored in asynchronous and SDLC modes.

Write Register 11

WR11 contains the clock mode control and directs the source of the transmit and receive clocks.


D7 D6 D5 D4 D3 D2 D1 D0
0 0 NOT VALID
0 1 TxC OUT=TRANSMIT CLOCK
1 0 TxC OUT=BR GENERATOR OUTPUT
1 1 TxC OUT= DPLL OUTPUT*
1
0 0 TRANSMIT CLOCK=RxC PIN
0 1 NOT VALID
1 0 TRANSMIT CLOCK=BR GENERATOR OUTPUT
1 1 TRANSMIT CLOCK = DPLL OUTPUT*
0 0 RECEIVE CLOCK=RxC PIN
0 1 NOT VALID
1 0 RECEIVE CLOCK=BR GENERATOR OUTPUT
1 1 RECEIVE CLOCK = DPLL OUTPUT*
0

*This feature is not presently supported by Pro-Log.

Figure 6-17. Write Register 11 Bit Map

Write Register 12

WR12 contains the lower byte of the baud rate generator time constant.

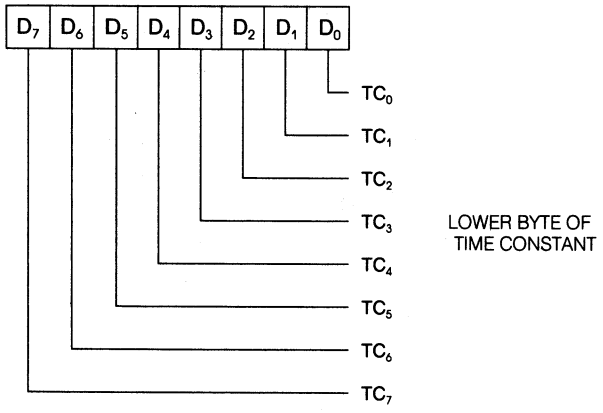


Figure 6-18. Write Register 12 Bit Map

D7 This bit must be reset for operation with the 7863.

D6-D5 Receive Clock

These bits determine the source of the receive clock. For normal asynchronous operations, the receive clock = baud rate generator output (D6D5=10). For normal synchronous operation, the receive clock = RxC pin (D6D5=00). Pro-Log does not presently support the receive clock = BR generator output (D6D5=10) feature on the 7863.

D4-D3 These bits determine the source of the transmit clock. For normal operation, D4D3=10. Pro-Log does not presently support the D4D3=11 feature on the 7863.

D2 This bit must be set for operation with the 7863.

D1-D0 Transmit Clock Output Source

These bits determine the signal output at the SCC transmit clock pins. For normal synchronous modes, D1D0=01. Pro-Log does not presently support the D1D0=11 feature on the 7863.

Write Register 13

WR13 contains the upper byte of the baud rate generator time constant.

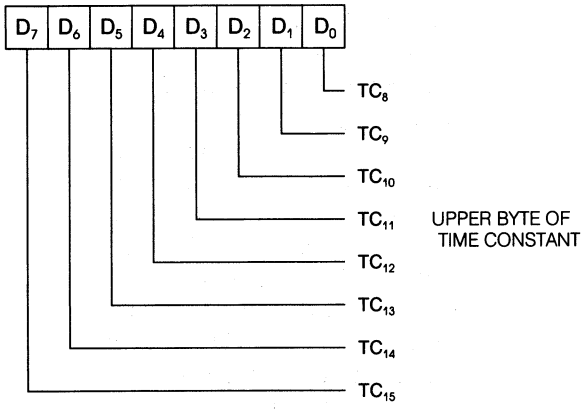
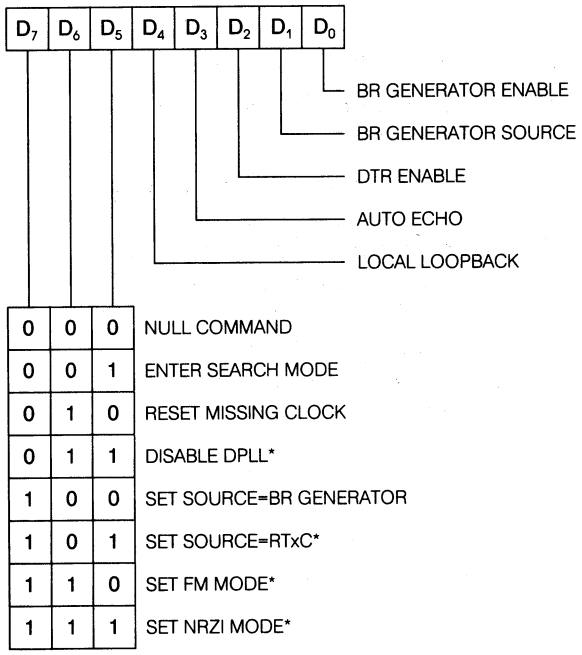


Figure 6-19. Write Register 13 Bit Map

Write Register 14

WR14 contains miscellaneous control bits: baud rate generator, phase-locked loop control, auto echo, and local loopback modes.


D7 D6 D5 D4 D3 D2 D1 D0
BR GENERATOR ENABLE
BR GENERATOR SOURCE
DTR ENABLE
AUTO ECHO
LOCAL LOOPBACK
0 0 0 NULL COMMAND
0 0 1 ENTER SEARCH MODE
0 1 0 RESET MISSING CLOCK
0 1 1 DISABLE DPLL*
1 0 0 SET SOURCE=BR GENERATOR
1 0 1 SET SOURCE=RTxC*
1 1 0 SET FM MODE*
1 1 1 SET NRZI MODE*

*This feature is not presently supported by Pro-Log.

Figure 6-20. Write Register 14 Bit Map

D7-D5 Digital Phase-Locked Loop Command Bits

These bits encode the eight commands for the digital phase-locked loop (DPLL). Pro-Log does not presently support these features on the 7863.

NULL CODE: This command has no effect on the DPLL.

ENTER SEARCH MODE: This command resets the clock missing latches in RR10 and arms the latches to detect the next missing clock edge. This is only used in the FM mode.

DISABLE DPLL: This command disables the DPLL, resets the clock missing latches in RR10, and forces a Search mode state.

SET SOURCE = RTxC: This command forces the DPLL clock to come from the RTxC pin.

SET FM MODE: This command directs the DPLL to recover the clock from FM or Manchester-encoded data.

SET NRZI MODE: This command directs the DPLL to operate in NRZI mode.

D4 Local Loopback

When this bit is set, the input to the receiver is internally connected to TXD, the transmitter output. Data received on RXD, the normal receiver input, is ignored. Transmitted data can then be read back through the receiver. The Auto Enable functions of CTS* and DCD* are disabled.

D3 Auto Echo

When this bit is set, the input to the transmitter is internally connected to RXD, the receiver input. Data written to the SCC transmit buffer is ignored. Received data is then automatically sent back out on TXD. The Auto Enable function of CTS* is disabled.

D2 DTR Enable

This bit should be reset to zero for the DTR pin to follow the state of the DTR bit in WR5.

D1 Baud Rate Generator Source

This bit selects the source of the clock for the baud rate generator. If this bit is set, the clock is the SCC's 4.9152 MHz PCLK input. If D1 is reset, the baud rate generator source is the receive clock input.

D0 Baud Rate Generator Enable

When this bit is set, the baud rate generator is enabled. With D0 reset, the baud rate generator is disabled.

Write Register 15

WR15 contains external/status interrupt response control.

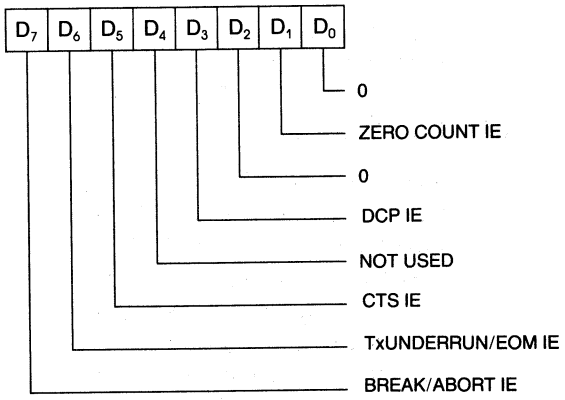


Figure 6-21. Write Register 15 Bit Map

D7 Break/Abort Interrupt Enable

When this bit is set, a change in the break/abort status of the receiver generates an external/status interrupt.

D6 Transmit Underrun/End of Message

When D6 is set, a change in the Tx Underrun/EOM latch in the receiver causes an external/status interrupt.

D5 CTS Interrupt Enable

When D5 is set, a change in the CTS pin causes an external/status interrupt. On channel B, CTS is driven by the 8087 math coprocessor interrupt output.

D4 This bit is not used on the 7863.

D3 DCD Interrupt Enable

When D3 is set, a change in the DCD pin causes an external/status interrupt. On channel B, DCD is driven by the STD BUS maskable interrupt, INTRQ*.

D2 This bit is not used and must be reset.

D1 Zero Count Interrupt Enable

When D1 is set, an external/status interrupt is generated whenever the counter in the baud rate generator reaches zero.

D0 This bit is not used and must be reset.

SCC STATUS REGISTERS

All SCC read registers require two bytes to be read except RR0. The first byte must be a write to WR0 with the register select code in the least significant four bits. The next byte is read from the selected read register. The byte following a read or write to any register is directed to register 0. Figure 6-22 lists the functions assigned to each status register. Following are detailed descriptions of each status register.

All references to pin states are as seen at the SCC. The state that appears at the pin is the state that is seen in the status bit. The RS-232-C interface (channel A) inverts the signal. Pro-Log presently does not support some features of the SCC. These are marked with an asterisk (*) in the register diagram.

In the bit descriptions, bits are referred to as being "set" or "reset." "Set" refers to bit DX=1 and "reset" refers to bit DX=0. All control and status registers apply to both SCCs.

RR0	Transmit/Receive buffer status and External status
RR1	Special Receive Condition status
RR2	Modified interrupt vector (channel B only) Unmodified interrupt vector (channel A only)
RR3	Interrupt Pending bits (channel A only)
RR8	Receive buffer
RR10	Miscellaneous status
RR12	Lower byte of baud rate generator time constant
RR13	Upper byte of baud rate generator time constant
RR15	External/Status interrupt information

Figure 6-22. Read Register Functions

Read Register 0

RR0 contains transmit and receive buffer status and external status.

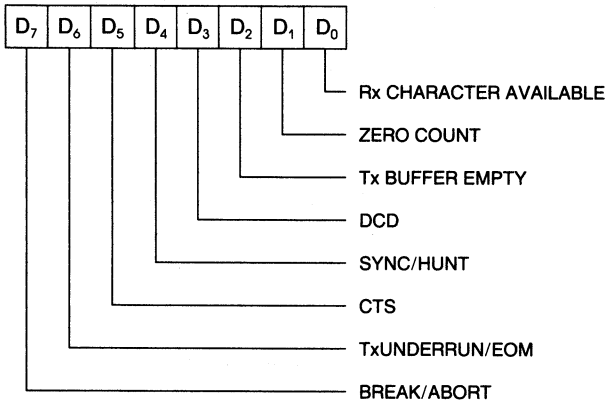


Figure 6-23. Read Register 0 Bit Map

D7 Break/Abort

In asynchronous modes, this bit is set when a "break" (continuous zeros) is detected. After the Reset external/status interrupts command (command 2, WR0) has been executed, the break bit is reset when the break stops. If external status interrupts have been enabled (D0, WR1), both the detection and termination of the break sequence causes an interrupt. An external/status vector is sent if the Status Affects Vector bit is programmed (D0, WR9). In SDLC mode, an abort sequence (7 or more ones) sets this bit. It is not used in other synchronous modes.

D6 Transmit Underrun/End of Message

When this bit has been reset by the "reset Tx Underrun/EOM Latch" command in WR0 and a transmit underrun occurs, an external/status interrupt is generated. Only a zero-to-one transition of this bit causes an interrupt.

D5 Clear to Send

This bit shows the status of the channel CTS* pin at the time of the first change of any of the five external/status bits (DCD, CTS, Sync/Hunt, Break/Abort, or Transmit Underrun/EOM) after power-on, channel reset, or an external/status interrupt (command 2). On channel B, CTS* is driven by the 8087 math coprocessor's interrupt output. Odd numbers of transitions on the CTS pin while another external/status interrupt is pending also cause an external/status interrupt condition. To get the current state of the CTS* pin, this bit must be read immediately following a reset external/status interrupt command (command 2, WR0). A transition at the CTS* pin causes an interrupt if external interrupts are enabled at D0 of WR1. An external/status vector is sent if the Status Affects Vector bit is programmed (D0, WR9).

D4 Sync/Hunt

In synchronous modes, this bit is reset when character synchronization is achieved and is set by writing the Enter Hunt Mode Bit. In asynchronous modes, this bit is not used.

D3 Data Carrier Detect

This bit shows the status of the DCD* pin at the time of the first change of any of the five external/status bits (DCD, CTS, Sync/Hunt, External Interrupt, Break/Abort, or Transmit Underrun/EOM) after power-on, channel reset, or an external/status interrupt (command 2). On channel B, DCD* is driven by the STD BUS signal INTRQ*. The change of any of these bits latches the status bits of RR0. To get the current state of the DCD* pin, this bit must be read immediately following a reset external/status interrupt command (command 2, WR0). Any odd number of DCD transitions while another external/status interrupt is pending also causes an external/status interrupt condition. A transition at the DCD* pin causes an interrupt if external interrupts are enabled at D0 of WR1. An external/status vector is sent if the Status Affects Vector bit is programmed (D0, WR9).

D2 Transmit Buffer Empty

This bit is set whenever the channel transmit buffer is empty except during CRC transmission in synchronous mode. This causes an interrupt that modifies the vector of RR2 if the Status Affects Vector bit is programmed (D0, WR9).

D1 Zero Count

This bit is set when the baud rate generator count is at zero. An external/status interrupt is generated if no other interrupts have been generated. If an interrupt

is pending, no interrupt is generated until the interrupt service is completed. If the zero count does not persist until the end of the interrupt under service routine, no interrupt is generated.

D0 Receive Character Available

This bit is set when a character is available in the receive buffer. The receiver ports are quadruply buffered (three storage registers and an input shift register) to allow additional time for the CPU to service an interrupt at the beginning of a block of high-speed data. This bit is reset automatically when the receive buffers are all empty. Error status reflects the current byte in the receive buffer. When the receive character bit is set, RR1 should be read and stored to avoid losing any error information. The data received can then be read. Error information does not need to be read with vectored interrupts, since a modified vector is generated if there is an error. A receive character is available to be read a maximum of 13 system clock cycles from the RxC signal that samples the last bit of the data. Since the receiver port has three storage registers, three characters can be contained in the SCC without a read. A fourth character received without a read causes an overrun error in the last receiver storage register.

Read Register 1

RR1 contains the Special Receive condition status, residue codes, and error conditions.

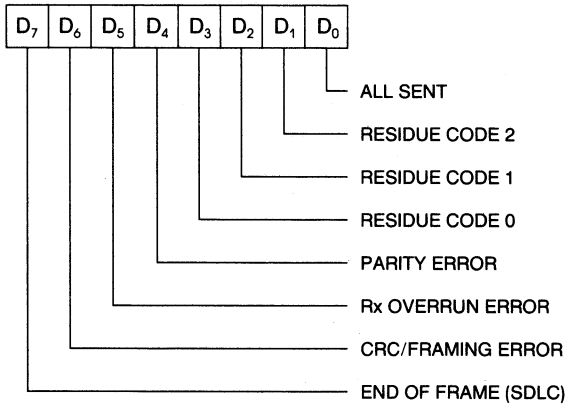


Figure 6-24. Read Register 1 Bit Map

D7 End of Frame (SDLC)

In SDLC mode, this bit indicates reception of a valid ending flag and validation of CRC error and residue codes.

D6 CRC/Framing Error

If a framing error occurs (in asynchronous modes), this bit is set (and not latched) only for the character on which it occurred. Detection of a framing error adds in additional half-bit time to the character time so that the framing error will not also be interpreted as a new start bit. In synchronous modes, this bit indicates the result of comparing the CRC to the appropriate value.

D5 Receiver Overrun Error

This indicates that more than three characters have been received without a read from the CPU. Only the character that has been written over is flagged with this error, but when this character is read, the error condition is latched until reset by the error reset command (command 6, WR0). If the Status Affects Vector bit is enabled (D0, WR9), the character that has been overrun interrupts with the special receive condition vector.

D4 Parity Error

When parity is enabled (D0, WR9), this bit is set for those characters whose parity does not match the programmed one (D1, WR4). The bit is latched so that once an error occurs, the bit remains set until the Error Reset command (command 6, WR0) is given.

D3-D1 Residue Codes

In the SDLC mode, these bits indicate the length of the information field, when the I field is not an integral multiple of the character length. These codes are meaningful upon transfer of the End of Frame bit. If the last character boundary coincides with the boundary of the I field and CRC field, the residue code is always 011.

D0 All Sent

This bit is set when all characters have completely cleared the transmitter (XMT Buffer Empty) in asynchronous modes. Interrupts are not caused by transitions of this bit. It is always set in synchronous modes.

Read Register 2

This register contains the interrupt vector as written into channel A WR2. When this register is accessed in channel A, the vector returned is the vector actually stored in WR2. When this register is accessed in channel B, the vector returned includes status information (the highest priority interrupt pending) in bits V1, V2, and V3 or bits V4, V5, and V6, depending upon the state of the Status High/Status Low bit in WR9 and independent of the state of the V15 bit in WR9. The vector is modified as shown in Figure 6-26. If no interrupts are pending, V6=1, V5=1, V4=0, or V3=0, V2=1, V1=1 and other bits are as programmed.

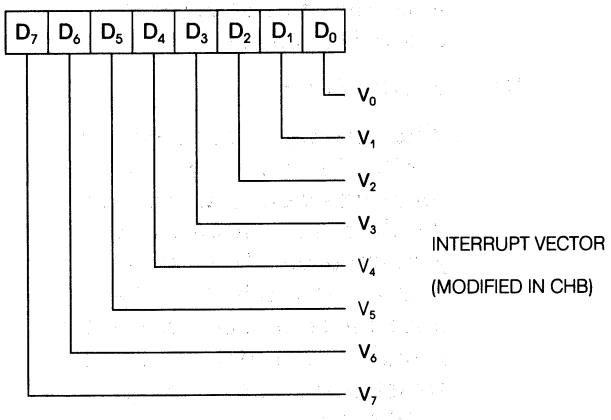


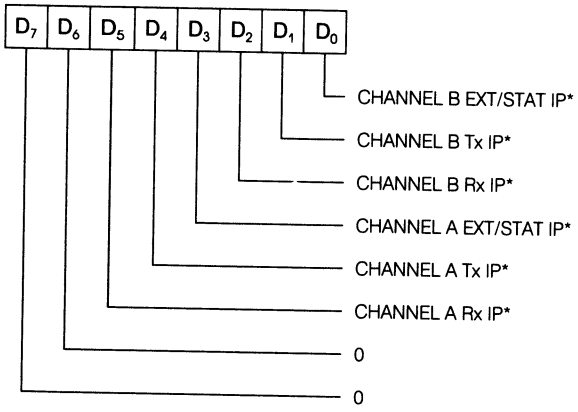
Figure 6-25. Read Register 2 Bit Map

V_3	V_2	V_1	Status High/Status Low = 0
V_4	V_5	V_6	Status High/Status Low = 1
0	0	0	Ch B transmit buffer Empty
0	0	1	Ch B External/Status Change
0	1	0	Ch B Receive Character Available
0	1	1	Ch B Special Receive Condition
1	0	0	Ch A transmit buffer Empty
1	0	1	Ch A External/Status Change
1	1	0	Ch A Receive Character Available
1	1	1	Ch A Special Receive Condition

Figure 6-26. Status Vector Mode Table

Read Register 3

RR3 contains the status of each interrupt pending bit (channel A only).



*ALWAYS 0 IN B CHANNEL

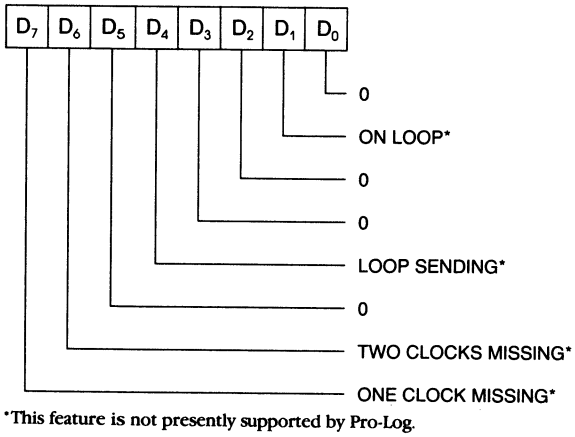
Figure 6-27. Read Register 3 Bit Map

Read Register 8

RR8 is the receive buffer.

Read Register 10

RR10 contains miscellaneous transmitter and receiver status parameters. These features are presently not supported by Pro-Log on the 7863.



*This feature is not presently supported by Pro-Log.

Figure 6-28. Read Register 10 Bit Map

D7 One Clock Missing

In the FM mode, this bit is set if one clock is missed. In NRZI mode and if the DPLL is disabled, this bit is always reset.

D6 Two Clocks Missing

In the FM mode, this bit is set when the DPLL misses two consecutive clock edges. In the NRZI mode, this bit is always reset.

D4 Loop Sending

In SDLC loop mode, this bit is set while the transmitter is transmitting on the loop.

D1 On Loop

In SDLC loop mode, this bit is set while the respective SCC is on the loop.

Read Register 12

RR12 contains the lower byte of the baud rate generator time constant as stored in WR12.

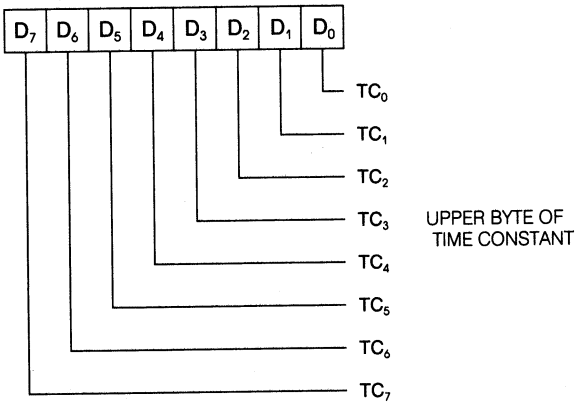


Figure 6-29. Read Register 12 Bit Map

Read Register 13

RR13 contains the upper byte of the baud rate generator time constant as stored in WR13.

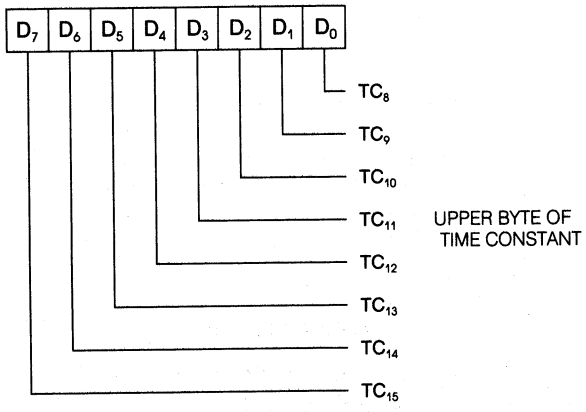


Figure 6-30. Read Register 13 Bit Map

Read Register 15

RR15 contains external/status interrupt control information as stored in WR15.

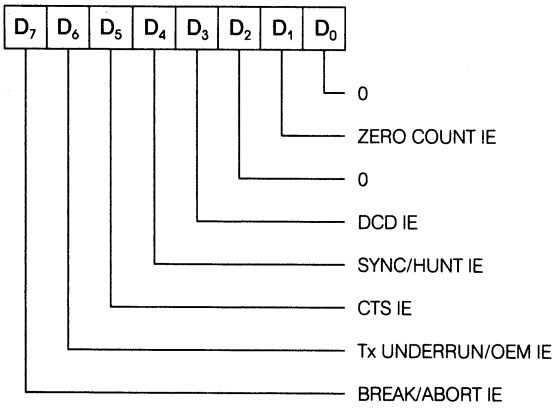


Figure 6-31. Read Register 15 Bit Map

SCC OPERATION

Initialization

The SCC initialization has three stages. In the first stage, the operation modes (i.e., bits per character, parity, etc.) are programmed and the constants (interrupt vector, time constants, etc.) are loaded. In the second stage, the hardware functions are enabled (transmitter, receiver, baud rate generator, etc.). Operating modes should be programmed before hardware functions are enabled. In the third stage, interrupts are enabled. Figure 6-32 shows the order in which SCC registers should be initialized.

REGISTER	DATA	COMMENTS
		Stage 1. Modes and constants
WR9	11000000	Hardware Reset
WR4	XXXXXXXX	Transmit/Receive Control selects Async or Sync Mode
WR2	XXXXXXXX	Program interrupt vector (optional)
WR3	XXXXXXX0	Selects receiver control. Bit D0 (Rx enable) must be set to 0 at this time
WR5	XXXX0XXX	Selects transmit control. Bit D3 (Tx enable) must be set to 0 at this time.
WR6	XXXXXXXX	Program sync characters
WR7	XXXXXXXX	Program sync characters
WR9	000X0XXX	Select interrupt control Bit D3 (Master interrupt enable) must be set to 0.
WR10	XXXXXXXX	Miscellaneous control (optional)
WR11	XXXXXXXX	Clock control
WR12	XXXXXXXX	Time constant lower byte
WR13	XXXXXXXX	Time constant upper byte
WR14	XXXXXXX0	Miscellaneous control. Bit D0 (BR Generator enable) must be set to 0 at this time.
WR14	XXXSSSSS	This register may require multiple writes if more than one command is used.
		Stage 2. Enables
WR3	SSSSSSS1	Set D0 (Rx enable)
WR5	SSSS1SSS	Set D3 (Tx enable)
WR0	10000000	Reset TxCRC
WR14	000SSSS1	BR Generator enable. Set Bit D0 (BR Generator enable)
		Stage 3. Interrupt Enables (optional)
WR15	XXXXX0X0	Enable external interrupts
WR0	00010000	Reset EXT/STATUS twice
WR0	00010000	Reset EXT/STATUS twice
WR1	SSSXXSXX	Enable receive, transmit, and external interrupt master
WR9	000SXSSS	Enable Master Interrupt Bit D3

1 (Set to one)
0 (Set to zero)
X (User's choice)
S (Same as previously programmed)

Figure 6-32. SCC Initialization Order

Baud Rate

The individual baud rates for each channel are selected by programming the WR12 and WR13 time constants for their respective channels. The formula for determining the appropriate time constant for given baud rate is shown below:

Time constant =

$$X1: \left[\frac{4.9152 \times 10^6}{2 \times (\text{Baud Rate})} \right] - 2$$

$$X2: \left[\frac{4.9152 \times 10^6}{32 \times (\text{Baud Rate})} \right] - 2$$

Figure 6-33 shows time constants to achieve common baud rates.

BAUD RATE	CONSTANT (x1 clock)		%	CONSTANT (x16 clock)		%
	Decimal	Hex	Error	Decimal	Hex	Error
19.2K*	126	007E	—0—	6	0006	—0—
9600	254	00FE	—0—	14	000E	—0—
4800	510	01FE	—0—	30	001E	—0—
2400	1022	03FE	—0—	64	0040	—0—
1800	1361	0551	0.17	83	0053	0.39
1200	2046	07FE	—0—	126	007E	—0—
600	4094	0FFE	—0—	254	00FE	—0—
300	8190	1FFE	—0—	510	01FE	—0—
200	12286	2FFE	—0—	766	02FE	—0—
150	16382	3FFF	—0—	1022	03FE	—0—
134.5	18270	475E	0.0007	1140	0474	0.0007
110	22340	5744	0.0008	1394	572	0.026
75	32766	7FFE	—0—	2046	07FE	—0—
50	49150	BFFE	—0—	3070	0BFE	—0—

*The maximum data rate according to the RS-232-C specification is 20K baud.

Figure 6-33. Time Constants for Common Baud Rates

Interrupts

The SCC has an internal interrupt priority chain. The priorities within the SCC are shown in Figure 6-34. The most common way to operate the SCC is to use the Status Affects Vector feature to minimize polling. SCC interrupts have a lower priority than 8088 internal or STD BUS interrupts.

HIGHEST PRIORITY	DESCRIPTION	VECTOR									
			V3	V2	V1			V6	V5	V4	
1	ChA RCV character or	XXXX	1	1	0	X	X	0	1	1	XXXX
	special RCV condition	XXXX	1	1	1	X	X	1	1	1	XXXX
2	ChA XMT Buffer Empty	XXXX	1	0	0	X	X	0	0	1	XXXX
3	ChA Status Change or External Interrupt	XXXX	1	0	1	X	X	1	0	1	XXXX
4	ChB RCV character or	XXXX	0	1	0	X	X	0	1	0	XXXX
	special RCV condition	XXXX	0	1	1	X	X	1	1	0	XXXX
5	ChB XMT Buffer Empty	XXXX	0	0	0	X	X	0	0	0	XXXX
6	ChB Status Change or External Interrupt	XXXX	0	0	1	X	X	1	0	0	XXXX

Including 8087 math coprocessor and STD BUS INTRQ interrupts.

Figure 6-34. SCC Internal Interrupt Priority

RS-232-C Handshaking (Channel A)

Three elements control handshaking for the RS-232-C port: the SCC, the processor, and the jumpers. The jumpers select to which RS-232-C line each handshake pin of the SCC is connected. The SCC can control some of the handshake lines and the processor can control all or some of them through the SCC registers, depending upon whether or not the auto-handshaking is enabled. Figure 6-35 shows which lines can be controlled by the SCC.

The CTS* and DCD* pins automatically enable/disable the transmitter and receiver if auto-handshaking is selected. If the receiver or transmitter is disabled while receiving/transmitting a character, the character may be lost. It is recommended that the character be repeated. Perform a channel reset command through WR9, re-initialize, and repeat the character.

If auto-handshaking is not enabled, the CTS* and DCD* pins must be monitored by the processor or they can generate interrupts if handshake interrupts are enabled. CTS* and DCD* are monitored through RR0. The DTR* and RTS* pins are controlled by the processor through WR5. There are no handshaking lines required on the RS-422 interface (channel B) so the CTS* and DCD* inputs are used to generate interrupts from the 8087 Math Coprocessor and STD BUS INTRQ*, respectively.

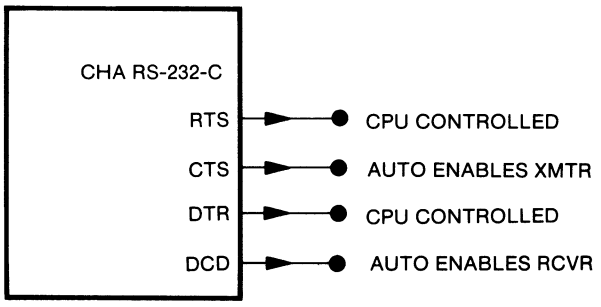


Figure 6-35. SCC RS-232-C Handshake Capability (Channel A)

Block Transfers

Block transfers can occur between the SCC and the CPU. Block transfers are controlled by the two wait/ready lines (W/RDY A* and W/RDY B*), one for each channel.

For block transfers, the wait/ready line should be programmed using bits 5 and 7 of WR1. The line is connected to the 8088 Ready/Clk Generator. The wait/ready line can be enabled at initialization and left enabled if all transfers will be block transfers. Since the line only becomes active when the processor attempts to read or write a character before the SCC is ready, it does not interfere with normal processor operation. Reading from or writing to the control or status registers is not affected by the wait/ready line.

The CPU block transfers are performed using the 8088 string instructions. Blocks of up to 128K bytes can be transferred. The Interrupt on First Character mode should be used when receiving in the block mode.

SECTION 7
OPERATING SOFTWARE

INTRODUCTION

This section contains hardware-level subroutine modules that initialize and operate the 7863 Serial Communications Controller (SCC). The software in this section can be used without license from Pro-Log. Although tested and believed correct, this software is not represented to be free from errors or copyright infringement, or appropriate for any specific application.

The subroutines are in Intel mnemonics and were generated on an IBM PC using Microsoft's Macroassembler. A copy of the source listing file is included at the end of this section, following the flowcharts.

DESCRIPTION

The software in this section is intended to show how the 7863 SCC can be initialized and operated and to provide subroutines that can be included in the user's applications program, if appropriate. This software displays various functions and modes of operation of the SCC. Channel A, the RS-232-C port, is used for asynchronous transmission and reception at 4800 baud; channel B, the RS-422-A port, is used for synchronous communications in SDLC format at 19.2K baud. There are 14 subroutines, most of which are subroutines within subroutines and are not addressed directly. Only seven of them need to be accessed directly to perform serial communications. Others may be useful as building blocks in your own software. All subroutines are listed in Figure 7-1.

If used as a package with user-supplied software as required, these subroutines can greatly simplify the use of the 7863 serial ports. This software is not complete in itself; for example, there is no software provided to process errors in data reception or changes in incoming handshake lines since these routines would be largely application dependent.

NAME	DESCRIPTION	MAIN SUBROUTINE	MAY BE USED BY ITSELF	NOT USEFUL BY ITSELF
(INIT COMM)	Initialize Communications Interface	X		
(GEN INIT TABLE)	Generate vector table, enable interrupts		X	
(INIT SCC A)	Initialize Channel A		X	
(INIT SCC B)	Initialize Channel B		X	
(XMT CH A ASYN)	Transmit asynchronously a block of data (Ch. A)	X		
(XMT CH B SDLC)	Transmit a block of data in SDLC format (Ch. B)	X		
(XMT CHAR)	Transmit a character			X
(END XMT?)	Verify that entire block has been transmitted			X
(RCV CHAR A)	Receive a character (Ch. A)	X		
(RCV CHAR B)	Receive a character (Ch. B)	X		
(SPCL RCV A)	Determine error (Ch. A)	X		
(SPCL RCV B)	Determine error/EOF (Ch. B)	X		

Figure 7-1. Subroutine Types

USER REQUIREMENTS

To use the software provided, certain requirements must be met by the user. The seven main subroutines and their individual requirements are listed in Figure 7-2. Those requirements and the following points should be considered before implementing the software:

* The user may wish to provide software for handshake control or for responding to changes in incoming handshaking lines.

* Because it is application dependent, no software is provided for response to errors in received data.

* The user may wish to provide a subroutine to store and/or respond to the received data. The data received is stored and a Jump to Subroutine instruction calls the user's own subroutines: (PROCESS DATA) asynchronous and (END OF FRAME) synchronous.

* This software operates with the 8088 vectored interrupt scheme. The interrupt type number can be specified by the user at the beginning of the listing (INTVECT EQU, interrupt type number). The (GEN INT TABLE) subroutine then writes the required lookup table entries in RAM at the low end of memory (00000-003FF hex). If ROM is located in this address block, the user must include the service routine addresses at the correct locations.

* The baud rates and other parameters selected by the SCC initialization routines should be checked to see that they match the application. The initialized values are given in the description of the (INIT COMM) routine.

* The number of bytes to be transferred and the address of the first data byte to be transferred must be in the appropriate mailboxes (as specified by the (XMT CH A ASYN) and (XMT CH B SDLC) subroutine specifications) before using the transmit routines.

* A buffer area in RAM must be defined for received character storage.

NAME	DESCRIPTION	USE REQUIREMENTS
(INIT COMM)	Initialize Communications Interface	None
(XMT CH A ASYN)	Transmit a block of data asynchronously (Ch. A.)	Data in memory, data pointer at XMT BUF A, number of bytes at XMT COUNT A.
(XMT CH B SDLC)	Transmit a block of data in SDLC format (Ch. B.)	Control and data in memory, data pointer at XMT BUF B, number of bytes at XMT COUNT B.
(RCV CHAR A)	Receive and store a character (Ch. A)	User provides software to process received data, receive buffer pointer at RCV BUF A.
(RCV CHAR B)	Receive and store a character (Ch. B)	Receive buffer pointer at RCV BUF B.
(SPCL RCV A)	Determine asynchronous transmission error (Ch. A)	User provides software to process errors
(SPCL RCV B)	Determine SDLC transmission error or end of frame (Ch. B)	User provides software to process errors and data frame

Figure 7-2. Main Subroutines Use Requirements

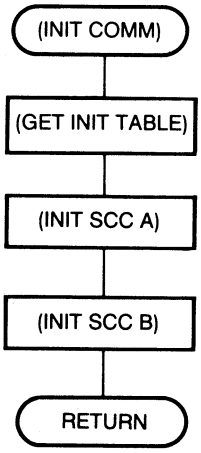


Figure 7-3. (INIT COM) Flowchart

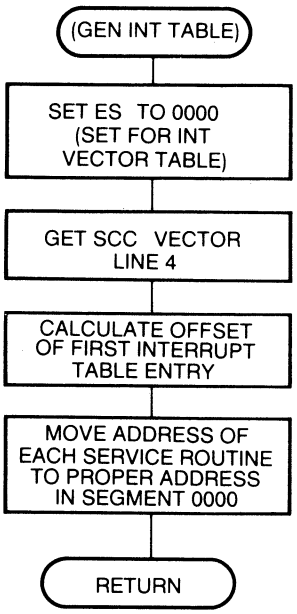

(GEN INT TABLE)
SET ES TO 0000 (SET FOR INT VECTOR TABLE)
GET SCC VECTOR LINE 4
CALCULATE OFFSET OF FIRST INTERRUPT TABLE ENTRY
MOVE ADDRESS OF EACH SERVICE ROUTINE TO PROPER ADDRESS IN SEGMENT 0000
RETURN

Figure 7-4. (GEN INT TABLE) Flowchart

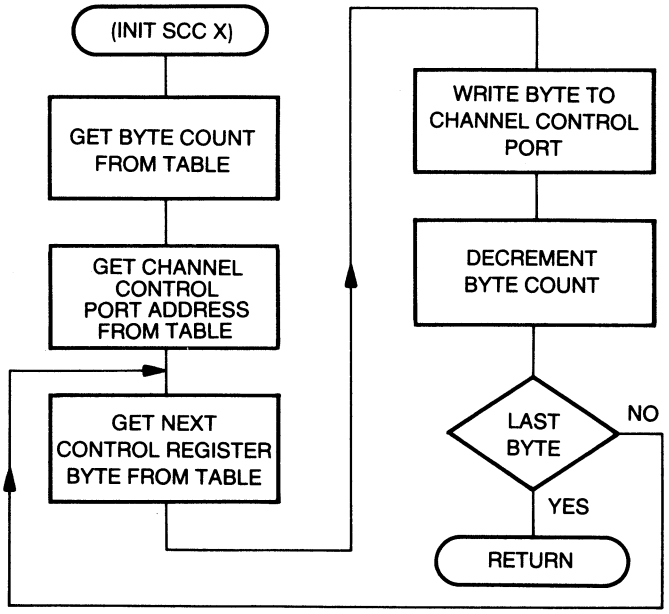


Figure 7-5. (INIT SCCA)/(INIT SCCB) Flowchart

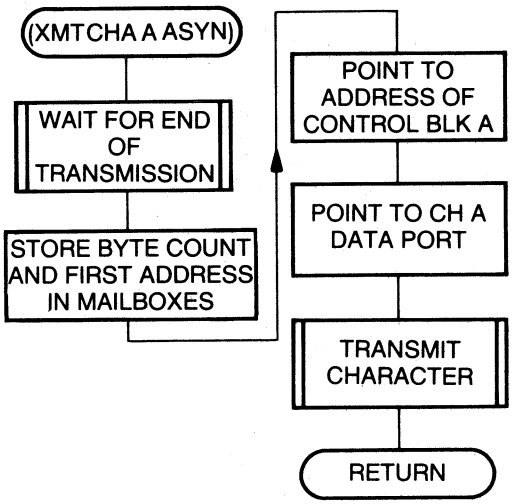

(XMTCHA A ASYN)
WAIT FOR END OF TRANSMISSION
STORE BYTE COUNT AND FIRST ADDRESS IN MAILBOXES
POINT TO ADDRESS OF CONTROL BLK A
POINT TO CH A DATA PORT
TRANSMIT CHARACTER
RETURN

Figure 7-6. (XMT CH A ASYN) Flowchart

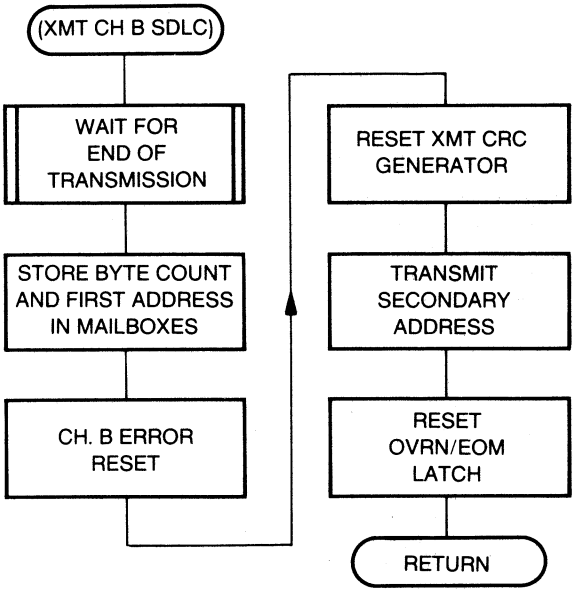

(XMT CH B SDLC)
WAIT FOR END OF TRANSMISSION
STORE BYTE COUNT AND FIRST ADDRESS IN MAILBOXES
CH. B ERROR RESET
RESET XMT CRC GENERATOR
TRANSMIT SECONDARY ADDRESS
RESET OVRN/EOM LATCH
RETURN

Figure 7-7. (XMT CH B SDLC) Flowchart

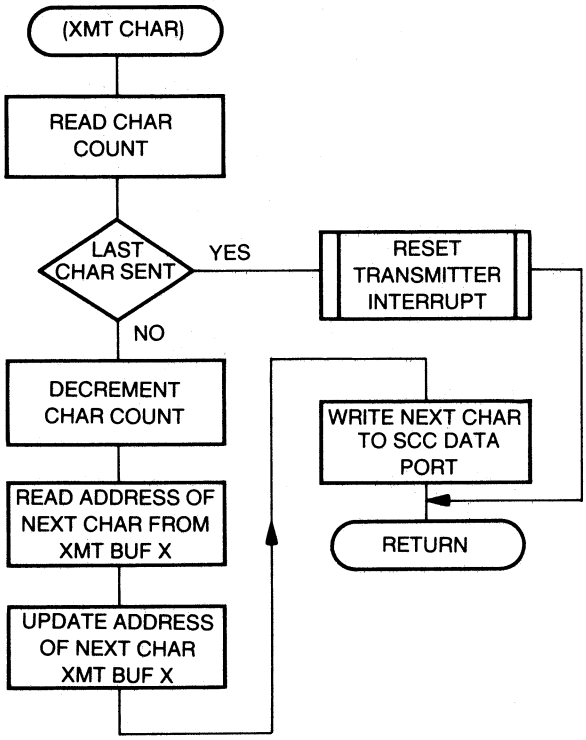


Figure 7-8. (XMT CHAR) Flowchart

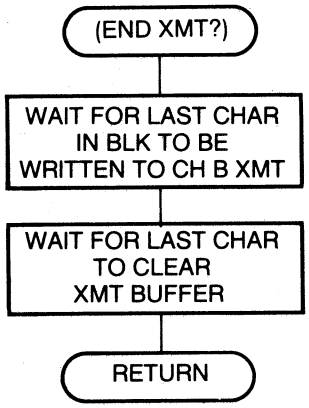


Figure 7-9. (END XMT?) Flowchart

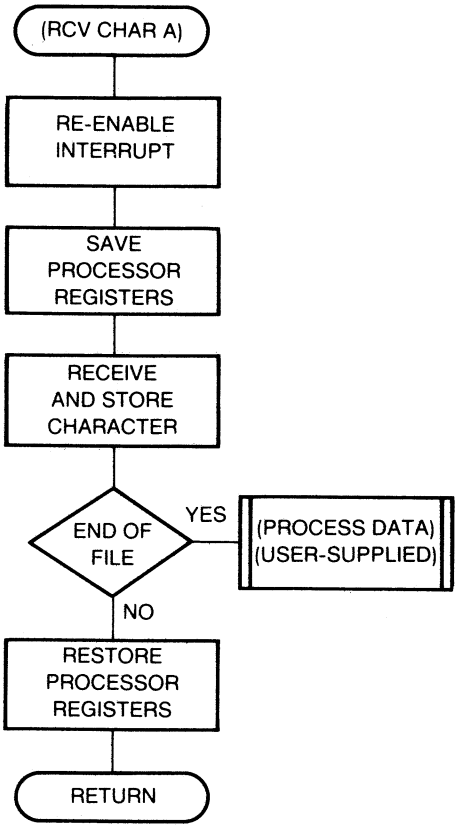


Figure 7-10. (RCV CHAR A) Flowchart

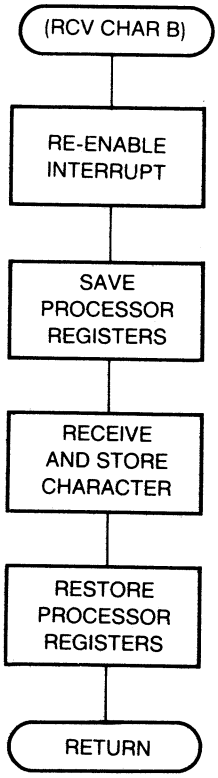


Figure 7-11 (RCV CHAR B) Flowchart

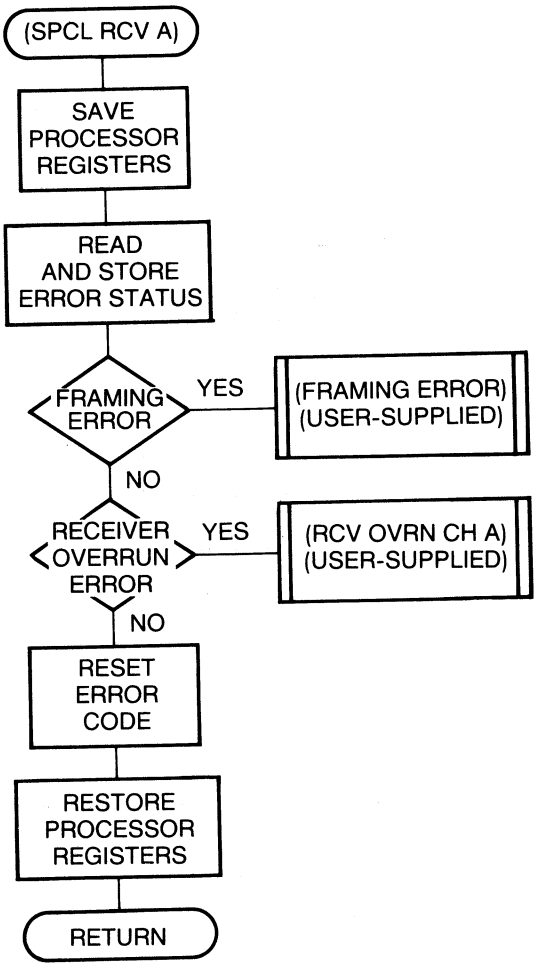


Figure 7-12. (SPCL RCV A) Flowchart

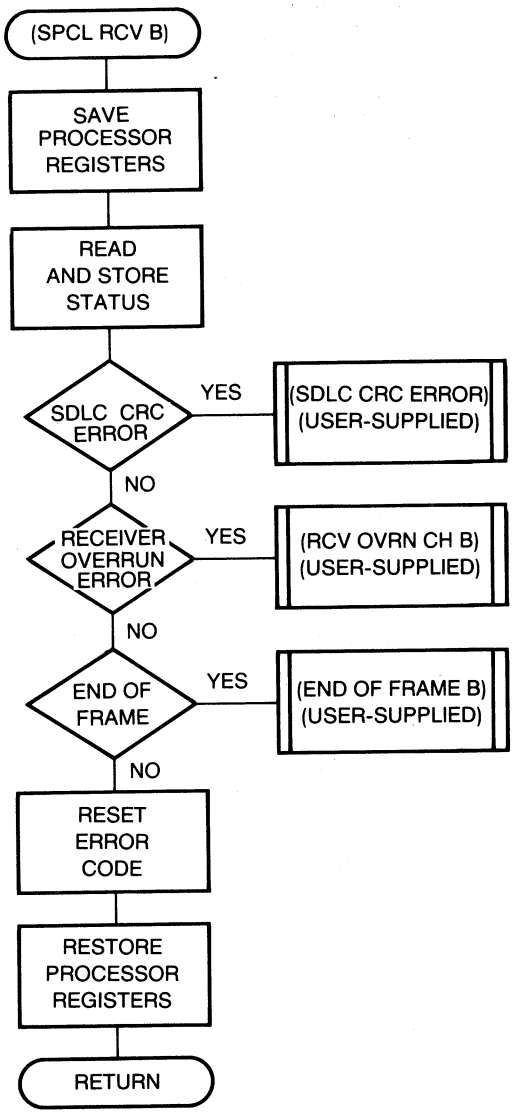


Figure 7-13 (SPCL RCV B) Flowchart

```
Microsoft MACRO Assembler  Version 3.00                                  06-13-85

                        PAGE    ,132

                        ;7863 SCC Drivers

0000                    CODE            SEGMENT PUBLIC
                                        ASSUME  CS:CODE

                        ;*****************************************************************************
                        ;* Interrupt type number = 240 (F0H).                                        *
                        ;* The user can specify a different type number if required.                 *
                        ;*****************************************************************************
= 00F0                  INTVECT         EQU     0F0H

                        ;*****************************************************************************
                        ;* These are the 7863 port addresses.                                        *
                        ;*****************************************************************************
= 8002                  SCC_CMD_A               EQU     08002H
= 8003                  SCC_DTA_A               EQU     08003H
= 8000                  SCC_CMD_B               EQU     08000H
= 8001                  SCC_DTA_B               EQU     08001H

                        ;*****************************************************************************
                        ;* The following memory locations are used as mailboxes.  Transmit and receive buffer    *
                        ;* pointers, error status bytes, and transmit byte counts are stored.        *
                        ;*****************************************************************************
0000  00                XMT_COUNT_A     DB      000H    ;Transmit byte count CH A
0001  0400              XMT_BUF_A       DW      0400H   ;Transmit buffer pointer CH A
0003  0500              RCV_BUF_A       DW      0500H   ;Receive buffer pointer CH A
0005  00                ERR_STAT_A      DB      000H    ;Error status CH A
0006  00                XMT_COUNT_B     DB      000H    ;Transmit byte count CH B
0007  0600              XMT_BUF_B       DW      0600H   ;Transmit buffer pointer CH B
0009  0700              RCV_BUF_B       DW      0700H   ;Receive buffer pointer CH B
000B  00                ERR_STAT_B      DB      000H    ;Error status CH B

                        ;*****************************************************************************
                        ;* INIT_SCC_A                                                                *
                        ;*         This subroutine initializes Channel A of the SCC with the data in the table   *
                        ;*         starting at INIT_DATA_A.                                          *
                        ;*                                                                           *
                        ;* Entry:  None                                                              *
                        ;* Exit:   SCC initialized, transmitter and receiver interrupts enabled      *
                        ;* Alters: AL, CX, DX, SI, Flags                                             *
                        ;* Calls:  None                                                              *
                        ;* Stack:  None                                                              *
                        ;*****************************************************************************
000C                    INIT_SCC_A      PROC    FAR             ;Jump past the table
000C  EB 24 90                          JMP     INIT_A
000F                    INIT_DATA_A:
000F  22                                DB      OFFSET INIT_DATA_A_END - OFFSET INIT_DATA_A-1
0010  09 80                            DB      09,080H         ;Register 9, Reset CH. A
0012  04 44                            DB      04,044H         ;Register 4, x16 Clk/1Stop/No Parity
0014  02 F0                            DB      02,INTVECT      ;Register 2, Interrupt Vector
0016  03 C0                            DB      03,0C0H         ;Register 3, Rcv 8 Bits, Rcv Disabled
0018  05 60                            DB      05,060H         ;Register 5, Xmt 8 Bits, Xmt Disabled
```

Figure 7-14. Program Listing

```
001A  09 01                          DB      09,001H         ;Register 9, Vector Includes Status, Status Lo
001C  0A 00                          DB      10,000H         ;Register 10, Misc Control
001E  0B 56                          DB      11,056H         ;Register 11, Rcv Clk=BR Gen, Xmt Clk=BR Gen
0020  0C 0E                          DB      12,00EH         ;Register 12, Low  Time Const. Byte (9600)
0022  0D 00                          DB      13,000H         ;Register 13, High Time Const. Byte (9600)
0024  0E 02                          DB      14,002H         ;Register 14, Baud Rate Gen=PCLK Input
0026  0F 00                          DB      15,000H         ;Register 15, Disable External Status Intr.
0028  03 C1                          DB      03,0C1H         ;Register 3, Rcv 8 Bits, Rcv Enabled
002A  05 EA                          DB      05,0EAH         ;Register 5, En DTR, En RTS, Xmt 8 BITS, En Xmt
002C  0E 03                          DB      14,003H         ;Register 14, BR Gen=PCLK Input, En BR Gen
002E  01 12                          DB      01,012H         ;Enable XMT/Rcv Interrupt
0030  09 09                          DB      09,09           ;Master Interrupt Enable
0032                          INIT_DATA_A_END:

0032  BE 000F R               INIT_A: MOV     SI,OFFSET INIT_DATA_A   ;Get table address
0035  2E: AC                          LODS    BYTE PTR INIT_DATA_A    ;Byte count to CX
0037  8A C8                          MOV     CL,AL
0039  32 ED                          XOR     CH,CH
003B  BA 8002                        MOV     DX,SCC_CMD_A            ;Port add to DX
003E                          INIT_A_LP:
003E  2E: AC                          LODS    BYTE PTR INIT_DATA_A    ;Get byte
0040  EE                            OUT     DX,AL           ;Output to SCC CH A
0041  E2 FB                          LOOP    INIT_A_LP               ;Repeat
0043  CB                            RET
0044                          INIT_SCC_A      ENDP

                              ;*****************************************************************************
                              ;* INIT_SCC_B                                                                *
                              ;*         This subroutine initializes Channel B of the SCC with the data in the table   *
                              ;*         starting at INIT_DATA_B.                                          *
                              ;*                                                                           *
                              ;* Entry:  None                                                              *
                              ;* Exit:   SCC initialized, transmitter and receiver interrupts enabled      *
                              ;* Alters: AL, CX, DX, SI, Flags                                             *
                              ;* Calls:  None                                                              *
                              ;* Stack:  None                                                              *
                              ;*****************************************************************************
0044                          INIT_SCC_B      PROC    FAR
0044  EB 2A 90                        JMP     INIT_B          ;Jump past the table
0047                          INIT_DATA_B:
0047  28                              DB      OFFSET INIT_DATA_B_END - OFFSET INIT_DATA_B-1
0048  09 40                          DB      09,040H         ;Register 9, Reset CH. B
004A  04 20                          DB      04,020H         ;Register 4, x1 Clk/SDLC/No Parity
004C  01 00                          DB      01,000H         ;Register 1, Interrupts Disabled
004E  02 F0                          DB      02,INTVECT      ;Register 2, Interrupt Vector
0050  03 CC                          DB      03,0CCH         ;Register  3, Rcv 8 Bits, Rcv Disabled
0052  05 60                          DB      05,060H         ;Register  5, Xmt 8 Bits, Xmt Disabled
0054  06 00                          DB      06,000H         ;Register  6, SDLC Address
0056  07 7E                          DB      07,07EH         ;Register  7, Sync Character
0058  0A 80                          DB      10,080H         ;Register 10, Misc Control
005A  0B 16                          DB      11,016H         ;Register 11, Rcv Clk=Xmt Clk, Xmt Clk=BR Gen
005C  0C 7E                          DB      12,07EH         ;Register 12, Low Time Const. Byte (19.2K)
005E  0D 00                          DB      13,000H         ;Register 13, High Time Const. Byte (19.2K)
0060  0E 02                          DB      14,002H         ;Register 14, Baud Rate Gen=PCLK Input
```

Figure 7-14. (Continued)

```
0062  0F 00                        DB      15,000H         ;Register 15, Disable External/Status Interrupts
0064  03 C1                        DB      03,0C1H         ;Register  3, Rcv 8 Bits, Rcv Enabled
0066  05 69                        DB      05,069H         ;Register  5, En DTR, En RTS, Xmt 8 Bits, En Xmt
0068  00 80                        DB      00,080H         ;Reset Xmt CRC
006A  0E 03                        DB      14,003H         ;Register 14, BR Gen=PCLK Input, En BR Gen
006C  01 12                        DB      01,012H         ;En Rcv and Xmt Interrupts
006E  09 09                        DB      09,09H          ;Master Int Enable
0070                               INIT_DATA_B_END:        ;End Of Table

0070  BE 0047 R          INIT_B: MOV     SI,OFFSET INIT_DATA_B   ;Get table address
0073  2E: AC                       LODS    BYTE PTR INIT_DATA_B    ;Byte count to CX
0075  8A C8                       MOV     CL,AL
0077  32 ED                       XOR     CH,CH
0079  BA 8000                     MOV     DX,SCC_CMD_B    ;Port add to DX
007C                     INIT_B_LP:
007C  2E: AC                       LODS    BYTE PTR INIT_DATA_B    ;Get byte
007E  EE                           OUT     DX,AL                   ;Output to SCC CH B
007F  E2 FB                        LOOP    INIT_B_LP                       ;Repeat
0081  CB                           RET
0082                     INIT_SCC_B      ENDP

                         ;*****************************************************************************************
                         ;* PROCESS_DATA                                                                          *
                         ;*        The user must supply this subroutine, which is called when a newline charactor*
                         ;*        is received by Channel A, indicating the end of an asynchronous message.      *
                         ;*****************************************************************************************
0082                     PROCESS_DATA    PROC FAR
0082  CB                           RET
0083                     PROCESS_DATA    ENDP

                         ;*****************************************************************************************
                         ;* XMT_CHAR                                                                              *
                         ;*        This subroutine is used by the XMT_CH_A_ASYN and XMT_CH_B_SDLC subroutines,    *
                         ;*        and by the transmitter empty interrupt service routines for both channels.     *
                         ;*                                                                                       *
                         ;* Entry:  SI = Address of mailbox where address of next character to be sent is stored. *
                         ;*        DI = Address of mailbox where character count is stored                        *
                         ;*        DX = SCC transmit buffer port address                                          *
                         ;* Exit:   Mailboxes: Address of next character incremented, character count decremented *
                         ;* Alters: AL, DI, DX, Flags                                                             *
                         ;* Calls:  None                                                                          *
                         ;* Stack:  None                                                                          *
                         ;*****************************************************************************************
0083                     XMT_CHAR                PROC FAR                ;SI=XMT_BUF_X offset, DI=XMT_COUNT_X offset, DX=SCC
                          port address
0083  B0 00                           MOV     AL,00H          ;Load byte count to AL
0085  2E: 3A 05                       CMP     AL,CS:[DI]              ;Last char sent?
0088  74 10                           JZ      LAST            ;Jump if it has been
008A  2E: FE 0D                       DEC     BYTE PTR CS:[DI]        ;Decrement byte count
008D  2E: 8B 3C                       MOV     DI,CS:[SI]      ;Output data byte
0090  2E: 8A 05                       MOV     AL,CS:[DI]
0093  EE                              OUT     DX,AL
0094  47                              INC     DI              ;Point to next data byte
0095  2E: 89 3C                       MOV     CS:[SI],DI
```

Figure 7-14. (Continued)

```
0098  4A                        DEC     DX              ;Point DX to SCC control port
0099  CB                        RET
009A  4A                LAST:   DEC     DX              ;Point DX to SCC control port
009B  B0 28                     MOV     AL,28H          ;Reset Xmt interrupt pending
009D  EE                        OUT     DX,AL
009E  CB                        RET
009F                    XMT_CHAR        ENDP

                        ;*****************************************************************************************
                        ;* END_XMT_A? and END_XMT_B?                                                             *
                        ;*         These subroutines test to see if previous transmissions from each of the two  *
                        ;*         SCC channels have been completed.  The program will not return from this      *
                        ;*         subroutine until the last character of the previous message has cleared the   *
                        ;*         transmit buffer.                                                              *
                        ;*                                                                                       *
                        ;* Entry:  None                                                                          *
                        ;* Exit:   Previous transmission completed                                               *
                        ;* Alters: AL, DX, Flags                                                                 *
                        ;* Calls:  None                                                                          *
                        ;* Stack:  None                                                                          *
                        ;*****************************************************************************************
009F                    END_XMT_A?      PROC FAR
009F  B0 00                     MOV     AL,00H          ;Wait for end of previous transmission
00A1  2E: 3A 06 0000 R  WAIT:   CMP     AL,XMT_COUNT_A
00A6  75 F9                     JNE     WAIT
00A8  BA 8002                   MOV     DX,SCC_CMD_A    ;Point to CH A command port
00AB  EC                NOTYET: IN      AL,DX           ;Wait for Xmt Buf empty
00AC  24 40                     AND     AL,40H
00AE  3C 40                     CMP     AL,40H
00B0  75 F9                     JNE     NOTYET
00B2  CB                        RET                     ;OK, return
00B3                    END_XMT_A?      ENDP

00B3                    END_XMT_B?      PROC FAR
00B3  B0 00                     MOV     AL,00H          ;Wait for end of previous transmission
00B5  2E: 3A 06 0006 R  WAIT2:  CMP     AL,XMT_COUNT_B
00BA  75 F9                     JNE     WAIT2
00BC  BA 8000                   MOV     DX,SCC_CMD_B    ;Point to CH B command port
00BF  EC                NOTYET2:        IN      AL,DX           ;Wait for Xmt Buf empty
00C0  24 40                     AND     AL,40H
00C2  3C 40                     CMP     AL,40H
00C4  75 F9                     JNE     NOTYET2
00C6  CB                        RET                     ;OK, return
00C7                    END_XMT_B?      ENDP

                        ;*****************************************************************************************
                        ;* XMT_CH_A_ASYN                                                                         *
                        ;*         This subroutine asynchronously transmits a block of data of up to 255 bytes.  *
                        ;*         The program first waits for any previous transmission to be completed.  Then  *
                        ;*         the number of bytes and the address of the block to be sent are moved to the  *
                        ;*         appropriate mailboxes, and the first byte is sent and the program exits.      *
                        ;*         Thereafter, whenever the transmitter indicates it is ready, via interrupt,    *
                        ;*         the transmit interrupt service routine sends another byte.  When the          *
                        ;*         transmitter interrupts after* sending the last byte, the transmitter interrupt*
```

Figure 7-14. (Continued)

```
                        ;*        is reset and no more data is sent.                                    *
                        ;*                                                                              *
                        ;* Entry:  AL: Number of bytes to be sent                                       *
                        ;*        SI: Address of first byte to be sent                                  *
                        ;*        Memory: Data to be sent                                               *
                        ;* Exit:   Byte count decremented and stored at (XMT_COUNT_A)                   *
                        ;*        Data pointer incremented and stored at (XMT_BUF_A)                    *
                        ;*        First byte sent                                                       *
                        ;* Alters: AL, DI, DX, SI, Flags                                               *
                        ;* Calls:  END_XMT_A?, XMT_CHAR                                                *
                        ;* Stack:  3 words                                                             *
                        ;*******************************************************************************
00C7                    XMT_CH_A_ASYN   PROC FAR                    ;AL=Byte count, SI=1st address
00C7  50                                PUSH AX
00C8  9A 009F ---- R                    CALL    END_XMT_A?          ;Wait for last xmission to end
00CD  58                                POP     AX
00CE  2E: A2 0000 R                     MOV     XMT_COUNT_A,AL      ;Put count and address in mailboxes
00D2  2E: 89 36 0001 R                  MOV     XMT_BUF_A,SI
00D7  BF 0000 R                         MOV     DI,OFFSET XMT_COUNT_A   ;Set up for XMT_CHAR subroutine
00DA  BE 0001 R                         MOV     SI,OFFSET XMT_BUF_A
00DD  BA 8003                           MOV     DX,SCC_DTA_A
00E0  9A 0083 ---- R                    CALL    XMT_CHAR            ;Transmit first char
00E5  CB                                RET
00E6                    XMT_CH_A_ASYN   ENDP

                        ;*******************************************************************************
                        ;* XMT_CH_B_SDLC                                                               *
                        ;*        This subroutine transmits a block of data using the SDLC format.  The program *
                        ;*        first waits for any previous transmission to be completed.  Then the transmit *
                        ;*        CRC generator is reset, the secondary address is sent, the reset overrun/end  *
                        ;*        of message latch is reset, and the program exits.  Thereafter, whenever the   *
                        ;*        transmitter indicates it is ready, via interrupt, the transmit interrupt      *
                        ;*        service routine sends another byte.  When the transmitter interrupts after    *
                        ;*        sending the last byte, the transmitter interrupt is reset and no more data is *
                        ;*        sent.                                                                 *
                        ;*                                                                              *
                        ;* Entry:  AL: Number of bytes to be sent                                       *
                        ;*        AH: SDLC secondary address                                            *
                        ;*        SI: Address of first byte to be sent                                  *
                        ;*        Memory: Data to be sent                                               *
                        ;* Exit:   Byte count stored at (XMT_COUNT_B)                                   *
                        ;*        Data pointer stored at (XMT_BUF_B)                                    *
                        ;*        SDLC address sent                                                     *
                        ;* Alters: AL, DI, DX, SI, Flags                                               *
                        ;* Calls:  END_XMT_B?, XMT_CHAR                                                *
                        ;* Stack:  3 words                                                             *
                        ;*******************************************************************************
00E6                    XMT_CH_B_SDLC   PROC FAR            ;AL=Byte count, AH=SDLC destination address, SI=Address of
                        first data
00E6  50                                PUSH AX
00E7  9A 00B3 ---- R                    CALL    END_XMT_B?      ;Wait for last xmission to end
00EC  58                                POP     AX
00ED  2E: A2 0006 R                     MOV     XMT_COUNT_B,AL  ;Put count and address in mailboxes
00F1  2E: 89 36 0007 R                  MOV     XMT_BUF_B,SI
```

Figure 7-14. (Continued)

```
00F6  BA 8000                          MOV     DX,SCC_CMD_B    ;Point to control reg CH B
00F9  B0 30                            MOV     AL,030H         ;Error reset
00FB  EE                               OUT     DX,AL
00FC  B0 80                            MOV     AL,080H         ;Reset Xmt CRC Gen
00FE  EE                               OUT     DX,AL
00FF  42                               INC     DX              ;Point to SCC data port
0100  8A C4                            MOV     AL,AH           ;Send SDLC address
0102  EE                               OUT     DX,AL
0103  4A                               DEC     DX              ;Point to SCC control port
0104  B0 C0                            MOV     AL,0C0H         ;Reset Xmt Ovrn/EOM latch
0106  EE                               OUT     DX,AL
0107  CB                               RET
0108                                   ENDP

                                       ;*****************************************************************************
                                       ;* XMT_A_INT, XMT_B_INT                                                      *
                                       ;*         These are the Transmit Buffer Empty interrupt service routines for Channels A *
                                       ;*         and B.  The next character of the message is moved to the transmit buffer,    *
                                       ;*         and the byte count and buffer address are adjusted.  If the last character    *
                                       ;*         has been sent, the transmitter interrupt is reset.                *
                                       ;*                                                                           *
                                       ;* Entry:  None                                                              *
                                       ;* Exit:   Byte count decremented and stored at (XMT_COUNT_A) or (XMT_COUNT_B)       *
                                       ;*         Data pointer incremented and stored at (XMT_BUF_A) or (XMT_BUF_B)         *
                                       ;*         Next byte sent or transmit interrupt reset.                       *
                                       ;* Alters: None                                                              *
                                       ;* Calls:  XMT_CHAR                                                          *
                                       ;* Stack:  12 words                                                          *
                                       ;*****************************************************************************
0108                                   XMT_A_INT       PROC FAR                        ;Chan A Xmt Int Service
0108  FB                                               STI                     ;Reenable interrupts
0109  50                                               PUSH    AX              ;Save registers
010A  57                                               PUSH    DI
010B  56                                               PUSH    SI
010C  52                                               PUSH    DX
010D  BF 0000 R                                        MOV     DI,OFFSET XMT_COUNT_A   ;Set up for XMT_CHAR subroutine
0110  BE 0001 R                                        MOV     SI,OFFSET XMT_BUF_A
0113  BA 8003                                         MOV     DX,SCC_DTA_A
0116  9A 0083 ---- R                                   CALL    XMT_CHAR                ;Transmit next character
011B  B0 38                                            MOV     AL,038H         ;Reset highest IUS
011D  EE                                               OUT     DX,AL
011E  5A                                               POP     DX              ;Restore and return
011F  5E                                               POP     SI
0120  5F                                               POP     DI
0121  58                                               POP     AX
0122  CF                                               IRET
0123                                   XMT_A_INT       ENDP

0123                                   XMT_B_INT       PROC FAR                        ;Chan B Xmt Int Service
0123  FB                                               STI                     ;Reenable interrupts
0124  50                                               PUSH    AX              ;Save registers
0125  57                                               PUSH    DI
0126  56                                               PUSH    SI
0127  52                                               PUSH    DX
```

Figure 7-14. (Continued)

```
0128  BF 0006 R                        MOV     DI,OFFSET XMT_COUNT_B   ;Set up for XMT_CHAR subroutine
012B  BE 0007 R                        MOV     SI,OFFSET XMT_BUF_B
012E  BA 8001                          MOV     DX,SCC_DTA_B
0131  9A 0083 ---- R                    CALL    XMT_CHAR                ;Transmit next character
0136  BO 38                            MOV     AL,038H         ;Reset highest IUS
0138  EE                              OUT     DX,AL
0139  5A                              POP     DX              ;Restore and return
013A  5E                              POP     SI
013B  5F                              POP     DI
013C  58                              POP     AX
013D  CF                              IRET
013E                          XMT_B_INT       ENDP

                              ;*****************************************************************************
                              ;* EXT_STAT_A, EXT_STAT_B (User supplied)
                              ;*         These interrupt service routines should process External/Status interrupts
                              ;*         from Channels A and B.
                              ;*****************************************************************************
013E                          EXT_STAT_A      PROC FAR
013E  CF                                      IRET
013F                          EXT_STAT_A      ENDP

013F                          EXT_STAT_B      PROC FAR
013F  CF                                      IRET
0140                          EXT_STAT_B      ENDP

                              ;*****************************************************************************
                              ;* RCV_CHAR_A
                              ;*         This is an interrupt service routine that receives an asynchronously
                              ;*         transmitted character and stores it in memory.  The user must supply a
                              ;*         subroutine (PROCESS_DATA), called when a newline character is received, which
                              ;*         interprets the received data string.  If a framing error, receiver overrun,
                              ;*         or parity error occours, a Special Receive interrupt is generated.
                              ;*
                              ;* Entry:  None
                              ;* Exit:   Character stored in buffer
                              ;*         Next storage address stored at (RCV_BUF_A).
                              ;* Alters: None
                              ;* Calls:  PROCESS_DATA (user-supplied)
                                        *
                              ;* Stack:  8 words
                              ;*****************************************************************************
0140                          RCV_CHAR_A      PROC    FAR
0140  FB                                     STI                     ;Reenable interrupts
0141  50                                     PUSH    AX              ;Save registers
0142  52                                     PUSH    DX
0143  57                                     PUSH    DI
0144  BA 8003                               MOV     DX,SCC_DTA_A    ;Point to CH A data port
0147  2E: 8B 3E 0003 R                       MOV     DI,RCV_BUF_A    ;Point DI to CH A Rcv Buffer
014C  EC                                     IN      AL,DX           ;Input data
014D  2E: 88 05                              MOV     CS:[DI],AL      ;Store received data in buffer
0150  47                                     INC     DI              ;Point to next location in buffer
0151  2E: 89 3E 0003 R                       MOV     RCV_BUF_A,DI    ;Store next buffer location
0156  3C 0A                                  CMP     AL,00AH         ;ASCII new line character?
```

Figure 7-14. (Continued)

```
0158  75 05                              JNE     CONTINUE                ;No - continue
015A  9A 0082 ---- R                      CALL    PROCESS_DATA    ;End of message, go process data
015F  BA 8002                 CONTINUE:   MOV     DX,SCC_CMD_A    ;Reset highest IUS
0162  B0 38                               MOV     AL,038H
0164  EE                                 OUT     DX,AL
0165  5F                                 POP     DI              ;Restore registers
0166  5A                                 POP     DX
0167  58                                 POP     AX
0168  CF                                 IRET
0169                          RCV_CHAR_A  ENDP

                              ;*****************************************************************************
                              ;* RCV_CHAR_B                                                                *
                              ;*         This is an interrupt service routine that receives and stores synchronously   *
                              ;*         transmitted SDLC characters.  (If errors are detected, or the final flag      *
                              ;*         received, a different interrupt service routine (SPCL_RCV_B) is called.)  The *
                              ;*         user must supply routines to process the received SDLC frame.                 *
                              ;*                                                                                       *
                              ;* Entry:  None                                                                          *
                              ;* Exit:   Character stored in buffer                                                    *
                              ;*         Next storage address stored at (RCV_BUF_B).                                   *
                              ;* Alters: None                                                                          *
                              ;* Calls:  None                                                                          *
                              ;* Stack:  3 words                                                                       *
                              ;*                                                                                       *
                              ;*****************************************************************************
0169                          RCV_CHAR_B  PROC    FAR
0169  FB                                 STI                     ;Reenable interrupts
016A  50                                 PUSH    AX              ;Save registers
016B  52                                 PUSH    DX
016C  57                                 PUSH    DI
016D  BA 8001                            MOV     DX,SCC_DTA_B    ;Point to CH B data port
0170  2E: 8B 3E 0009 R                   MOV     DI,RCV_BUF_B    ;Point DI to CH B Rcv Buffer
0175  EC                                 IN      AL,DX           ;Input data
0176  2E: 88 05                          MOV     CS:[DI],AL      ;Store received data in buffer
0179  47                                 INC     DI              ;Point to next location in buffer
017A  2E: 89 3E 0009 R                   MOV     RCV_BUF_B,DI    ;Store next buffer location
017F  BA 8000                            MOV     DX,SCC_CMD_B    ;Reset highest IUS
0182  B0 38                               MOV     AL,038H
0184  EE                                 OUT     DX,AL
0185  5F                                 POP     DI              ;Restore registers
0186  5A                                 POP     DX
0187  58                                 POP     AX
0188  CF                                 IRET
0189                          RCV_CHAR_B  ENDP

                              ;*****************************************************************************
                              ;* FRAM_ERR_A, OVRN_ERR_A, OVRN_ERR_B, CRC_ERR_B, END_OF_FRAME_B (All user-supplied)     *
                              ;*         These subroutines are called by the SPCL_RCV_A and SPCL_RCV_B interrupt       *
                              ;*         service routines in response to various error conditions, and to an SDLC end- *
                              ;*         of-frame (Channel B).                                                         *
                              ;*****************************************************************************
0189                          FRAM_ERR_A  PROC FAR
0189  CB                                 RET
```

Figure 7-14. (Continued)

```
018A                        FRAM_ERR_A      ENDP

018A                        OVRN_ERR_A      PROC FAR
018A  CB                                    RET
018B                        OVRN_ERR_A      ENDP

018B                        OVRN_ERR_B      PROC FAR
018B  CB                                    RET
018C                        OVRN_ERR_B      ENDP

018C                        CRC_ERR_B               PROC FAR
018C  CB                                    RET
018D                        CRC_ERR_B               ENDP

018D                        END_OF_FRAME_B  PROC FAR
018D  CB                                    RET
018E                        END_OF_FRAME_B  ENDP

                            ;*****************************************************************************************
                            ;* SPCL_RCV_A                                                                            *
                            ;*         This is an interrupt service routine that indicates Channel A receiver errors.*
                            ;*         If a framing error or receiver overrun occours, the SCC generates a Special   *
                            ;*         Receive vectored interrupt which calls this routine.  The user must provide   *
                            ;*         the subroutines to process the detected error.  After the error is processed, *
                            ;*         the error code is reset.                                                      *
                            ;*                                                                                       *
                            ;* Entry:  None                                                                          *
                            ;* Exit:   Channel A RR1 status stored at (ERR_STAT_A)                                   *
                            ;* Alters: None                                                                          *
                            ;* Calls:  FRAM_ERR_A, OVRN_ERR_B (user-supplied)                                        *
                            ;* Stack:  4 words                                                                       *
                            ;*****************************************************************************************
018E                        SPCL_RCV_A      PROC FAR
018E  FB                                    STI                     ;Reenable interrupts
018F  50                                    PUSH    AX              ;Save registers
0190  52                                    PUSH    DX
0191  BA 8002                              MOV     DX,SCC_CMD_A    ;Point to SCC CH A, RR1
0194  B0 01                                 MOV     AL,001H
0196  EE                                    OUT     DX,AL
0197  EC                                    IN      AL,DX           ;Read error status
0198  2E: A2 0005 R                         MOV     ERR_STAT_A,AL   ;Store it
019C  24 40                                 AND     AL,040H         ;Framing error ?
019E  74 05                                 JZ      GO_ON
01A0  9A 0189 ---- R                        CALL    FRAM_ERR_A      ;Yes
01A5  2E: A0 0005 R             GO_ON:  MOV     AL,ERR_STAT_A   ;Overrun error ?
01A9  24 20                                 AND AL,020H
01AB  74 05                                 JZ      GO_ON2
01AD  9A 018A ---- R                        CALL    OVRN_ERR_A      ;Yes
01B2  B0 30                     GO_ON2: MOV     AL,030H         ;Error reset
01B4  EE                                    OUT     DX,AL
01B5  B0 38                                 MOV     AL,038H         ;Reset highest IUS
01B7  EE                                    OUT     DX,AL
01B8  5A                                    POP     DX              ;Restore and return
01B9  58                                    POP     AX
```

Figure 7-14. (Continued)

```
01BA  CF                                    IRET
01BB                              SPCL_RCV_A    ENDP

                                  ;*****************************************************************************
                                  ;* SPCL_RCV_B                                                                *
                                  ;*        This is an interrupt service routine that indicates an SDLC CRC or receiver   *
                                  ;*        overrun error, or end of frame.  The user must supply subroutines to process  *
                                  ;*        the errors and interpret the data received.  After processing, the error code *
                                  ;*        is reset.                                                          *
                                  ;*                                                                           *
                                  ;* Entry:  None                                                              *
                                  ;* Exit:   Channel B RR1 status stored at (ERR_STAT_B)                       *
                                  ;* Alters: None                                                              *
                                  ;* Calls:  CRC_ERR_B, OVRN_ERR_B, END_OF_FRAME_B (user-supplied)             *
                                  ;* Stack:  4 words                                                           *
                                  ;*****************************************************************************
01BB                              SPCL_RCV_B    PROC FAR
01BB  FB                                        STI                     ;Reenable interrupts
01BC  50                                        PUSH    AX              ;Save registers
01BD  52                                        PUSH    DX
01BE  BA 8000                                   MOV     DX,SCC_CMD_B    ;Point to SCC CH B, RR1
01C1  B0 01                                     MOV     AL,001H
01C3  EE                                        OUT     DX,AL
01C4  EC                                        IN      AL,DX           ;Read error status
01C5  2E: A2 000B R                             MOV     ERR_STAT_B,AL   ;Store it
01C9  24 40                                     AND     AL,040H         ;CRC error ?
01CB  74 05                                     JZ      GO_ON3
01CD  9A 018C ---- R                            CALL    CRC_ERR_B               ;Yes
01D2  2E: A0 000B R               GO_ON3: MOV   AL,ERR_STAT_B   ;Overrun error ?
01D6  24 20                                     AND AL,020H
01D8  74 05                                     JZ      GO_ON4
01DA  9A 018B ---- R                            CALL    OVRN_ERR_B      ;Yes
01DF  2E: A0 000B R               GO_ON4: MOV   AL,ERR_STAT_B   ;End of frame ?
01E3  24 80                                     AND AL,080H
01E5  74 05                                     JZ      GO_ON5
01E7  9A 018D ---- R                            CALL    END_OF_FRAME_B  ;Yes
01EC  B0 30                       GO_ON5: MOV   AL,030H         ;Error reset
01EE  EE                                        OUT     DX,AL
01EF  B0 38                                     MOV     AL,038H ;Reset highest IUS
01F1  EE                                        OUT     DX,AL
01F2  5A                                        POP     DX              ;Restore and return
01F3  58                                        POP     AX
01F4  CF                                        IRET
01F5                              SPCL_RCV_B    ENDP

                                  ;*****************************************************************************
                                  ;* GEN_INT_TABLE                                                             *
                                  ;*        This subroutine generates the interrupt vector table needed to service SCC    *
                                  ;*        interrupts and moves it into RAM at the proper location between 00000 and     *
                                  ;*        003FFH.  The location depends on the interrupt type number specified at the   *
                                  ;*        beginning of this program.                                         *
                                  ;*                                                                           *
                                  ;* Entry:  RAM for vector table, 32 bytes between 00000 and 003FFH, depending on         *
                                  ;*        interrupt type number.                                             *
```

Figure 7-14. (Continued)

```
                        ;* Exit:   Vector table in RAM                                                    *
                        ;* Alters: AX, DI, Flags                                                          *
                        ;* Calls:  None                                                                   *
                        ;* Stack:  1 word                                                                 *
                        ;*********************************************************************************
01F5                    GEN_INT_TABLE   PROC FAR
01F5  06                                PUSH    ES              ;Save extra segment
01F6  BF 0000                           MOV     DI,0000H                ;Set ES to 0000H
01F9  8E C7                             MOV     ES,DI
01FB  32 E4                             XOR     AH,AH           ;Calculate first table address
01FD  B0 F0                             MOV     AL,INTVECT
01FF  24 F0                             AND     AL,0F0H
0201  F8                                CLC
0202  D1 D0                             RCL     AX,1
0204  D1 D0                             RCL     AX,1
0206  8B F8                             MOV     DI,AX           ;Move table add to DI
0208  26: C7 05 0123 R                  MOV     WORD PTR ES:[DI],OFFSET XMT_B_INT       ;Move int svc add to table
020D  83 C7 02                          ADD     DI,2
0210  26: 8C 0D                         MOV     ES:[DI],CS      ;Move int svc code seg
0213  83 C7 06                          ADD     DI,6
0216  26: C7 05 013F R                  MOV     WORD PTR ES:[DI],OFFSET EXT_STAT_B      ;Next add
021B  83 C7 02                          ADD     DI,2
021E  26: 8C 0D                         MOV     ES:[DI],CS
0221  83 C7 06                          ADD     DI,6
0224  26: C7 05 0169 R                  MOV     WORD PTR ES:[DI],OFFSET RCV_CHAR_B      ;Next add
0229  83 C7 02                          ADD     DI,2
022C  26: 8C 0D                         MOV     ES:[DI],CS
022F  83 C7 06                          ADD     DI,6
0232  26: C7 05 01BB R                  MOV     WORD PTR ES:[DI],OFFSET SPCL_RCV_B      ;Next add
0237  83 C7 02                          ADD     DI,2
023A  26: 8C 0D                         MOV     ES:[DI],CS
023D  83 C7 06                          ADD     DI,6
0240  26: C7 05 0108 R                  MOV     WORD PTR ES:[DI],OFFSET XMT_A_INT       ;Next add
0245  83 C7 02                          ADD     DI,2
0248  26: 8C 0D                         MOV     ES:[DI],CS
024B  83 C7 06                          ADD     DI,6
024E  26: C7 05 013E R                  MOV     WORD PTR ES:[DI],OFFSET EXT_STAT_A      ;Next add
0253  83 C7 02                          ADD     DI,2
0256  26: 8C 0D                         MOV     ES:[DI],CS
0259  83 C7 06                          ADD     DI,6
025C  26: C7 05 0140 R                  MOV     WORD PTR ES:[DI],OFFSET RCV_CHAR_A      ;Next add
0261  83 C7 02                          ADD     DI,2
0264  26: 8C 0D                         MOV     ES:[DI],CS
0267  83 C7 06                          ADD     DI,6
026A  26: C7 05 018E R                  MOV     WORD PTR ES:[DI],OFFSET SPCL_RCV_A      ;Last add
026F  83 C7 02                          ADD     DI,2
0272  26: 8C 0D                         MOV     ES:[DI],CS
0275  07                                POP     ES              ;Restore ES
0276  CB                               RET
0277                    GEN_INT_TABLE   ENDP

                        ;*********************************************************************************
                        ;* INIT_COMM                                                                      *
                        ;*         This subroutine initializes SCC Channels A and B, generates the interrupt     *
```

Figure 7-14. (Continued)

```
                        ;*        vector table, and enables interrupts.  The initialized conditions should be   *
                        ;*        checked to be sure they match the application.  The baud rates are 4800 for   *
                        ;*        Channel A and 19.2K for Channel B.  Channel A (the RS-232-C channel) uses     *
                        ;*        asynchronous protocol, while Channel B (the RS-422 channel) uses synchronous  *
                        ;*        SDLC format.                                                                  *
                        ;*                                                                                      *
                        ;* Entry:  None                                                                         *
                        ;* Exit:   SCC initialized, vector table in RAM, interrupts enabled                     *
                        ;* Alters: AX, DX, DI, Flags
                        ;*
                        ;* Calls:  INIT_SCC_A, INIT_SCC_B, GEN_INT_TABLE                                       *
                        ;* Stack:  4 words                                                                     *
                        ;***************************************************************************************
0277                    INIT_COMM       PROC FAR
0277  9A 000C ---- R                    CALL    INIT_SCC_A
027C  9A 0044 ---- R                    CALL    INIT_SCC_B
0281  9A 01F5 ---- R                    CALL    GEN_INT_TABLE
0286  FB                               STI
0287  CB                               RET
0288                    INIT_COMM       ENDP

0288                    CODE    ENDS
                        END
```

Figure 7-14. (Continued)

```
Segments and Groups:

                N a m e                 Size    Align   Combine Class

CODE . . . . . .                        0288    PARA    PUBLIC

Symbols:

                N a m e                 Type    Value   Attr

CONTINUE . . . . . .                    L NEAR  015F    CODE
CRC_ERR_B. . . . . .                    F PROC  018C    CODE    Length =0001
END_OF_FRAME_B . . . . . .              F PROC  018D    CODE    Length =0001
END_XMT_A? . . . . . .                  F PROC  009F    CODE    Length =0014
END_XMT_B? . . . . . .                  F PROC  00B3    CODE    Length =0014
ERR_STAT_A . . . . . .                  L BYTE  0005    CODE
ERR_STAT_B . . . . . .                  L BYTE  000B    CODE
EXT_STAT_A . . . . . .                  F PROC  013E    CODE    Length =0001
EXT_STAT_B . . . . . .                  F PROC  013F    CODE    Length =0001
FRAM_ERR_A . . . . . .                  F PROC  0189    CODE    Length =0001
GEN_INT_TABLE. . . . . .                F PROC  01F5    CODE    Length =0082
GO_ON. . . . . .                        L NEAR  01A5    CODE
GO_ON2 . . . . . .                      L NEAR  01B2    CODE
GO_ON3 . . . . . .                      L NEAR  01D2    CODE
GO_ON4 . . . . . .                      L NEAR  01DF    CODE
GO_ON5 . . . . . .                      L NEAR  01EC    CODE
INIT_A . . . . . .                      L NEAR  0032    CODE
INIT_A_LP. . . . . .                    L NEAR  003E    CODE
INIT_B . . . . . .                      L NEAR  0070    CODE
INIT_B_LP. . . . . .                    L NEAR  007C    CODE
INIT_COMM. . . . . .                    F PROC  0277    CODE    Length =0011
INIT_DATA_A. . . . . .                  L NEAR  000F    CODE
INIT_DATA_A_END. . . . . .              L NEAR  0032    CODE
INIT_DATA_B. . . . . .                  L NEAR  0047    CODE
INIT_DATA_B_END. . . . . .              L NEAR  0070    CODE
INIT_SCC_A . . . . . .                  F PROC  000C    CODE    Length =0038
INIT_SCC_B . . . . . .                  F PROC  0044    CODE    Length =003E
INTVECT. . . . . .                      Number  00F0
LAST . . . . . .                        L NEAR  009A    CODE
NOTYET . . . . . .                      L NEAR  00AB    CODE
NOTYET2. . . . . .                      L NEAR  00BF    CODE
OVRN_ERR_A . . . . . .                  F PROC  018A    CODE    Length =0001
OVRN_ERR_B . . . . . .                  F PROC  018B    CODE    Length =0001
PROCESS_DATA . . . . . .                F PROC  0082    CODE    Length =0001
RCV_BUF_A. . . . . .                    L WORD  0003    CODE
RCV_BUF_B. . . . . .                    L WORD  0009    CODE
RCV_CHAR_A . . . . . .                  F PROC  0140    CODE    Length =0029
RCV_CHAR_B . . . . . .                  F PROC  0169    CODE    Length =0020
SCC_CMD_A. . . . . .                    Number  8002
SCC_CMD_B. . . . . .                    Number  8000
SCC_DTA_A. . . . . .                    Number  8003
SCC_DTA_B. . . . . .                    Number  8001
SPCL_RCV_A . . . . . .                  F PROC  018E    CODE    Length =002D
SPCL_RCV_B . . . . . .                  F PROC  01BB    CODE    Length =003A
```

Figure 7-14. (Continued)

```
WAIT . . . . . .    L NEAR  00A1    CODE
WAIT2. . . . . .    L NEAR  00B5    CODE
XMT_A_INT. . . . .  F PROC  0108    CODE    Length =001B
XMT_BUF_A. . . . .  L WORD  0001    CODE
XMT_BUF_B. . . . .  L WORD  0007    CODE
XMT_B_INT. . . . .  F PROC  0123    CODE    Length =001B
XMT_CHAR . . . . .  F PROC  0083    CODE    Length =001C
XMT_CH_A_ASYN. . .  F PROC  00C7    CODE    Length =001F
XMT_CH_B_SDLC. . .  F PROC  00E6    CODE    Length =0022
XMT_COUNT_A. . . .  L BYTE  0000    CODE
XMT_COUNT_B. . . .  L BYTE  0006    CODE

48504 Bytes free

Warning Severe
Errors  Errors
0       0
```

Figure 7-14. (Continued)

REFERENCE DRAWINGS

The schematic (Figure 8-1) and assembly drawing (Figure 8-2) in the following pages are included in this manual FOR REFERENCE USE ONLY. These drawings may differ in some respects from the card you receive from Pro-Log. If you require the specific revision documents of the schematic and assembly for the board shipped, please call Pro-Log Customer Support at (800) 538-9570, extension 3617. In California, call (408) 646-3617 from 7 a.m. to 4 p.m. (Pacific Time).

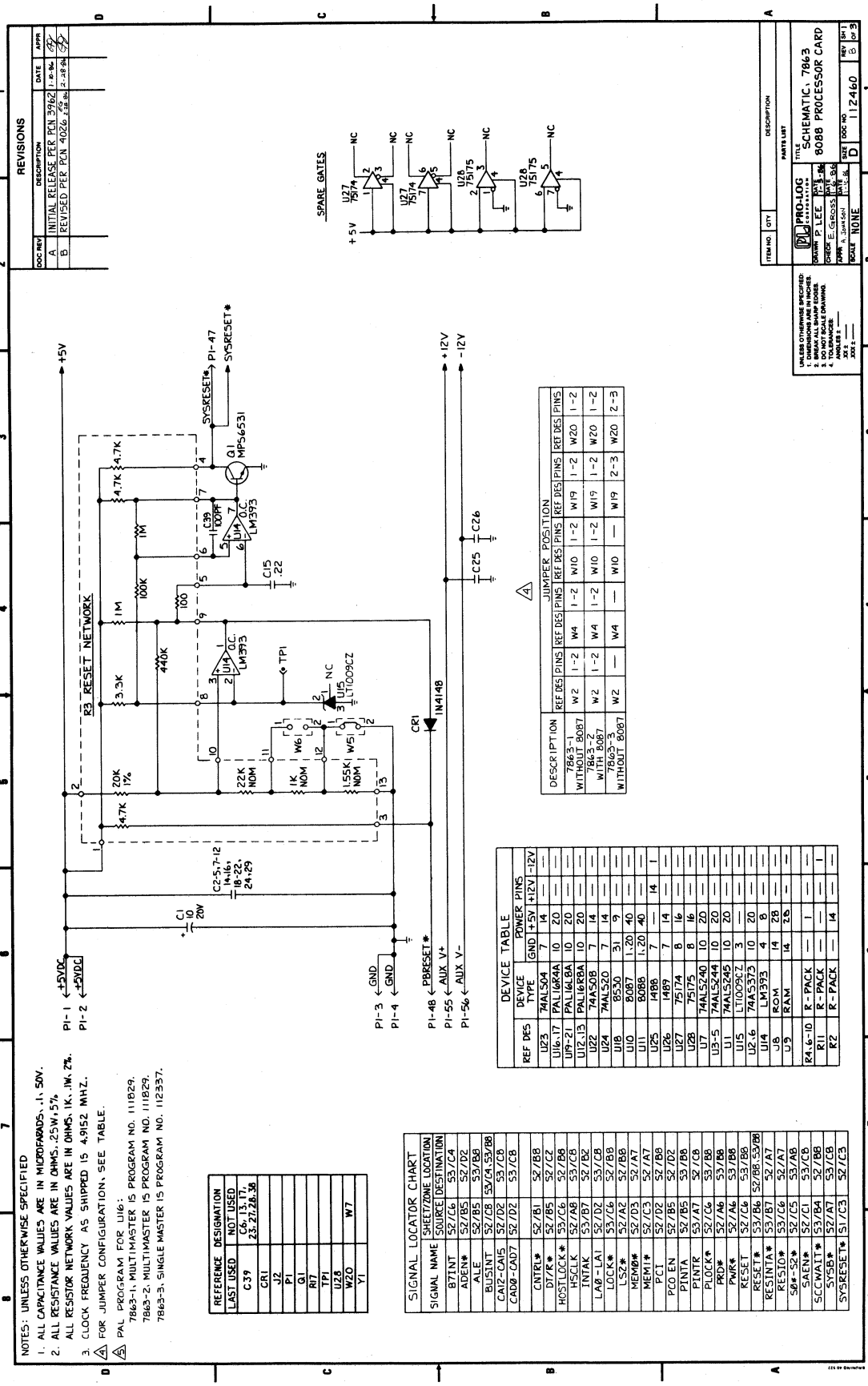

NOTES: UNLESS OTHERWISE SPECIFIED
1. ALL CAPACITANCE VALUES ARE IN MICROFARADS, .1, 50V.
2. ALL RESISTANCE VALUES ARE IN OHMS, .25W, 5%
ALL RESISTOR NETWORK VALUES ARE IN OHMS, 1K, 1W, 2%.
3. CLOCK FREQUENCY AS SHIPPED IS 4.9152 MHZ.
4. FOR JUMPER CONFIGURATION, SEE TABLE.
5. PAL PROGRAM FOR U16:
7863-1, MULTIMASTER IS PROGRAM NO. 111829.
7863-2, MULTIMASTER IS PROGRAM NO. 111829.
7863-3, SINGLE MASTER IS PROGRAM NO. 112337.
REVISIONS
DESCRIPTION
INITIAL RELEASE PER PCN 3962
REVISED PER PCN 4026
SCHEMATIC, 7863
8088 PROCESSOR CARD
PRO-LOG
DRAWN P. LEE
CHECK E. GROSS
APPR. A. SANTORI
SCALE NONE
SIZE D
DOC NO 112460
REV B
SHT 1 of 3
R3 RESET NETWORK
SPARE GATES
+5V
+12V
-12V
SYSRESET*
PBRESET*
AUX V+
AUX V-
GND
TPI
CR1
IN4148
Q1 MPS6531
U4 LM393
U5 LTI009CZ
U26 75175
U27 75174
U28 75175
C1 10 20V
C2-5,7-12 14-16, 18-22, 24-29
C15 .22
C25
C26
C39 100PF
20K 1%
4.7K
22K NOM
1K NOM
155K NOM
3.3K
440K
100K
1M
100
W1
W2
W4
W5
W10
W19
W20
P1-1 +5VDC
P1-2 +5VDC
P1-3 GND
P1-4 GND
P1-47 SYSRESET*
P1-48 PBRESET*
P1-55 AUX V+
P1-56 AUX V-
REFERENCE DESIGNATION
LAST USED NOT USED
C39 C6, 15, 17, 23, 27, 28, 38
CR1
J2
P1
Q1
R17
TP1
U28
W20
Y1
DEVICE TABLE
REF DES | DEVICE TYPE | POWER PINS GND | +5V | +12V | -12V
U23 | 74ALS04 | 7 | 14 | — | —
U16, 17 | PAL16R4A | 10 | 20 | — | —
U19-21 | PAL16L8A | 10 | 20 | — | —
U12, 13 | PAL16R8A | 10 | 20 | — | —
U22 | 74ALS08 | 7 | 14 | — | —
U24 | 74ALS20 | 7 | 14 | — | —
U18 | 8530 | 31 | 9 | — | —
U10 | 8087 | 1, 20 | 40 | — | —
U11 | 8088 | 1, 20 | 40 | — | —
U25 | 1498 | 7 | — | 14 | 1
U26 | 1489 | 7 | 14 | — | —
U27 | 75174 | 8 | 16 | — | —
U28 | 75175 | 8 | 16 | — | —
U7 | 74ALS240 | 10 | 20 | — | —
U3-5 | 74ALS244 | 10 | 20 | — | —
U1 | 74ALS245 | 10 | 20 | — | —
U15 | LTI009CZ | 3 | — | — | —
U2, 6 | 74ALS373 | 10 | 20 | — | —
U14 | LM393 | 4 | 8 | — | —
U8 | ROM | 14 | 28 | — | —
U9 | RAM | 14 | 28 | — | —
R4, 6-10 | R-PACK | — | 1 | — | —
R11 | R-PACK | — | — | 1 | —
R2 | R-PACK | — | 14 | — | —
JUMPER POSITION
DESCRIPTION | REF DES | PINS | REF DES | PINS | REF DES | PINS | REF DES | PINS
7863-1 WITHOUT 8087 | W2 | 1-2 | W4 | 1-2 | W10 | 1-2 | W19 | 1-2 | W20 | 1-2
7863-2 WITH 8087 | W2 | 1-2 | W4 | 1-2 | W10 | 1-2 | W19 | 1-2 | W20 | 1-2
7863-3 WITHOUT 8087 | W2 | — | W4 | — | W10 | — | W19 | 2-3 | W20 | 2-3
SIGNAL LOCATOR CHART
SIGNAL NAME | SHEET/ZONE LOCATION SOURCE | DESTINATION
B7INT | 52/C6 | 53/C4
ADEN* | 52/B5 | 52/D2
ALE | 52/B5 | 53/B8
BUSINT | 52/C8 | 53/C4, 53/B8
CAI2-CAI5 | 52/D2 | 53/C6
CAD0-CAD7 | 52/D2 | 53/C8
CNTRL* | 52/B1 | 52/B8
DT/R* | 52/B5 | 52/C2
HOSTLOCK* | 53/C6 | 52/B8
HSCLK | 52/A8 | 53/C8
INTAK | 53/B7 | 52/B2
LA0-LA1 | 52/D2 | 53/C8
LOCK* | 53/C6 | 52/B8
LS2* | 52/A2 | 52/B8
MEMO* | 52/D3 | 52/A7
MEMI* | 52/C3 | 52/A7
PCI | 52/D2 | 52/B8
PCO EN | 52/B5 | 52/D2
PINTA | 52/B5 | 53/B6
PINTR | 53/A7 | 52/C8
PLOCK* | 52/C6 | 53/B8
PRD* | 52/A6 | 53/B8
PWR* | 52/A6 | 53/B8
RESET | 52/C6 | 53/B8
RESET* | 53/B6 | 52/B8, 53/B8
RESINTA* | 53/B7 | 52/A7
RESIO* | 53/C6 | 52/A7
S0*-S2* | 52/C5 | 53/A8
SAEN* | 52/C1 | 53/C8
SCCWAIT* | 53/B4 | 52/B8
SYSB* | 52/A7 | 53/C8
SYSRESET* | 51/C3 | 52/C3

Figure 8-1. Schematic (Page 1 of 3)
FOR REFERENCE ONLY

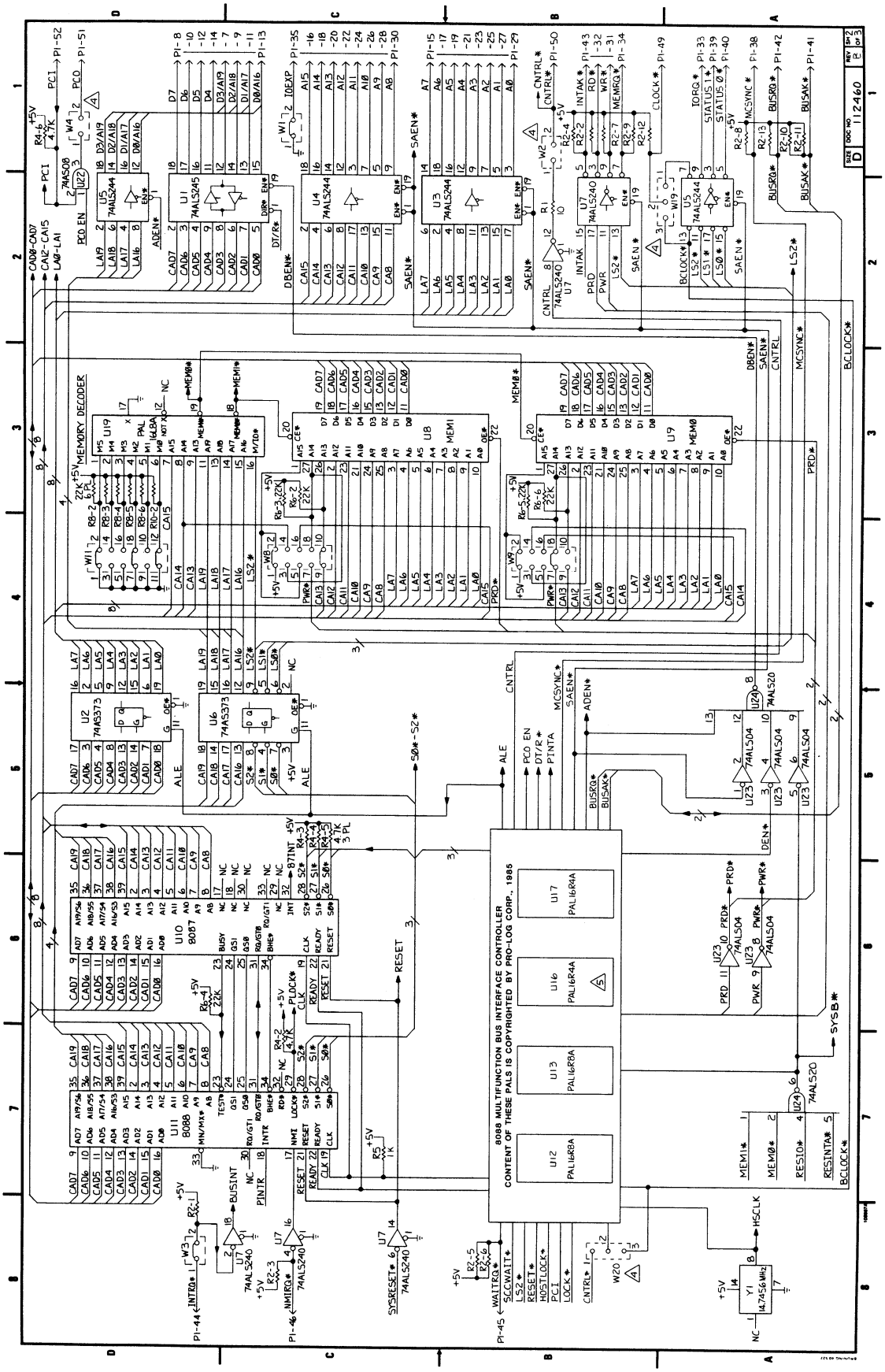


Figure 8-1. Schematic (Page 2 of 3)
FOR REFERENCE ONLY

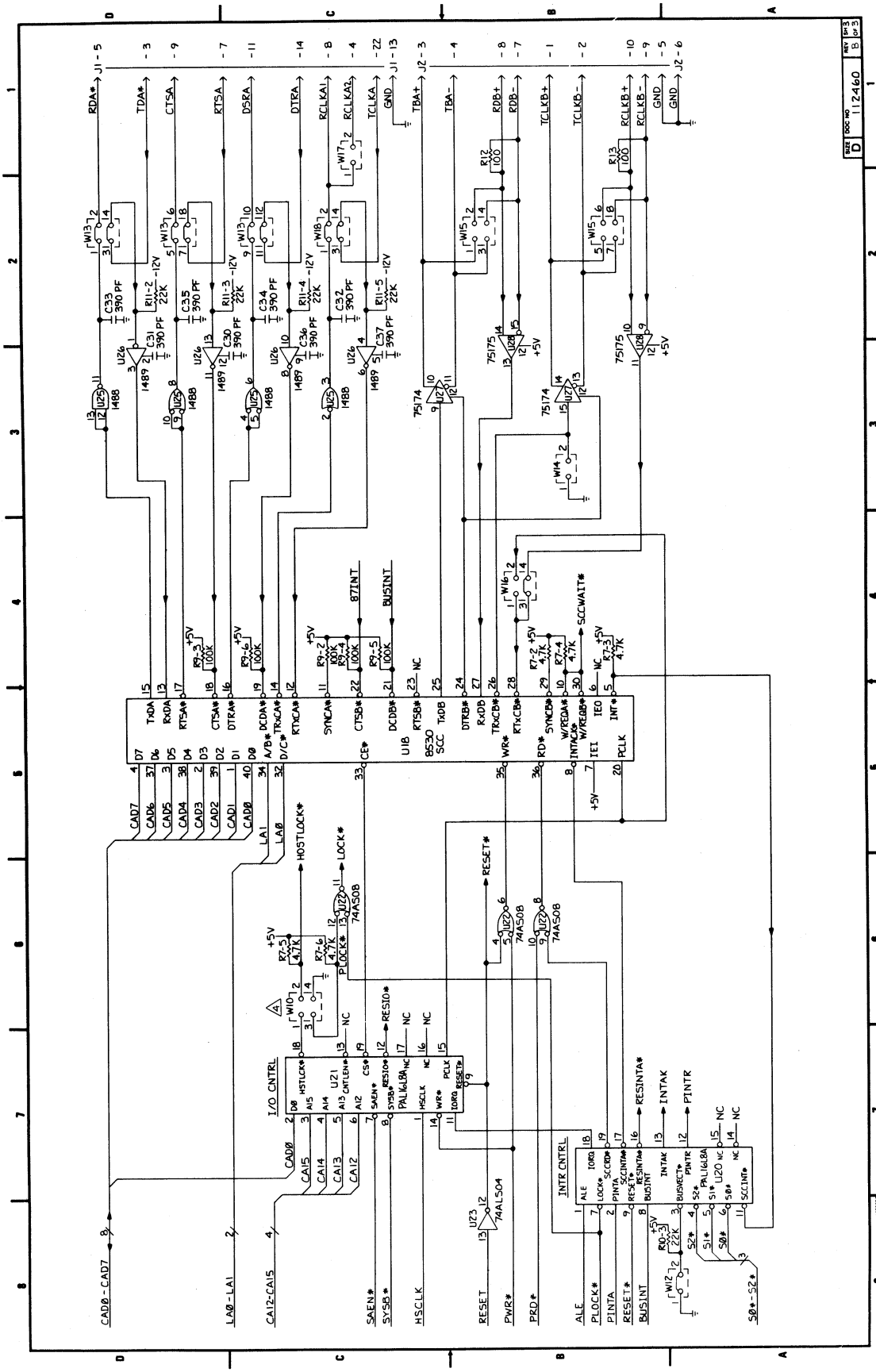


Figure 8-1. Schematic (Page 3 of 3)
FOR REFERENCE ONLY

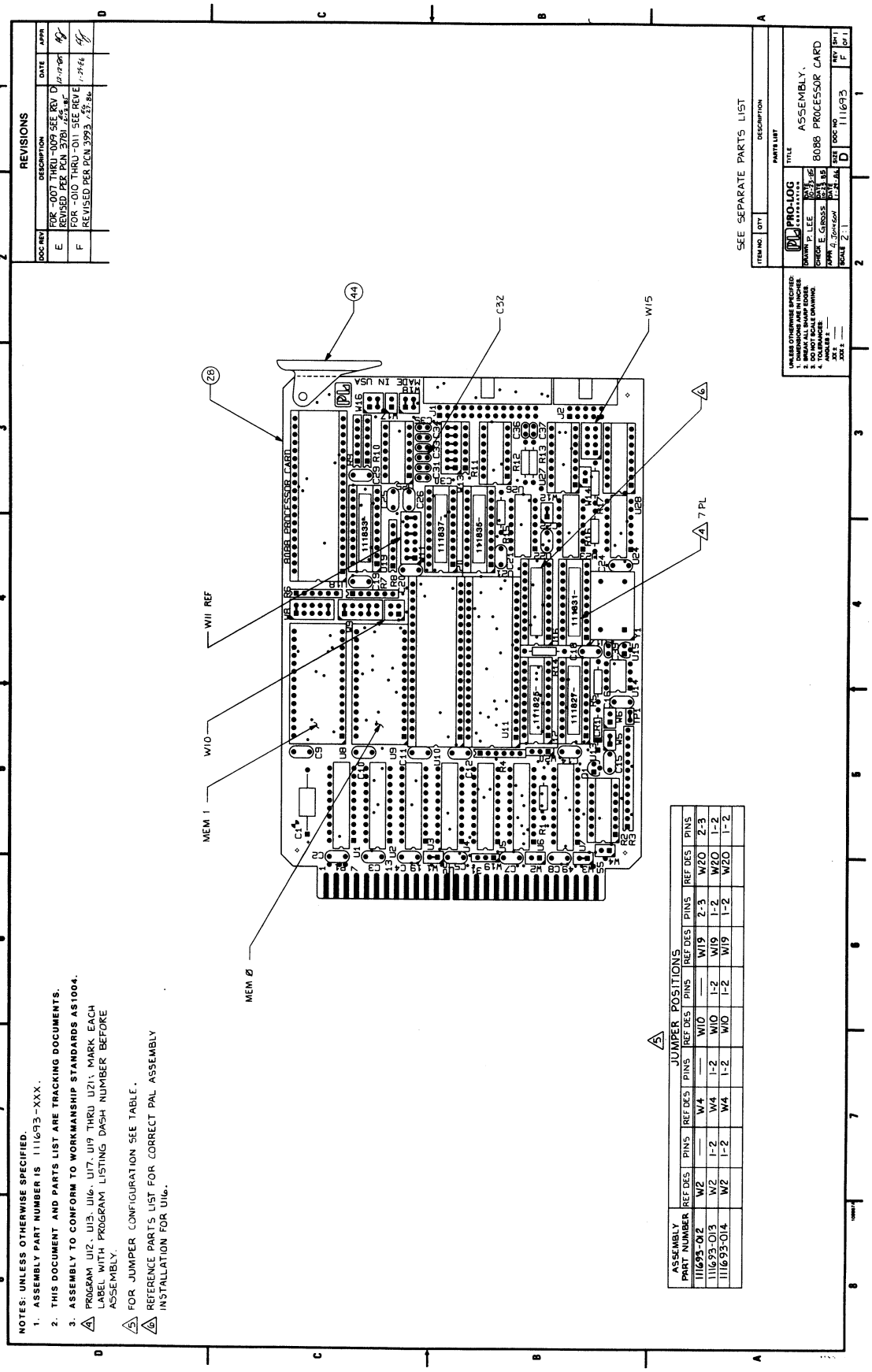


Figure 8-2. Assembly Drawing
FOR REFERENCE ONLY

RETURN FOR REPAIR PROCEDURES

Domestic Customers:

1. Call Customer Support at (800) 538-9570, ext. 3617. In California call (408) 646-3617 from 7 a.m. to 4 p.m. (Pacific Time).

2. Explain the problem and we may be able to solve it on the phone. If not, we will give you a Customer Return Order (CRO) number.

3. Please be sure to enclose a packing slip with CRO number, serial number of the equipment, if applicable, and reason for the return, with the name and telephone number of the person we should contact (preferably the user), if we have any further questions.

4. Pack the equipment in a solid cardboard box secured with packing material.

CAUTION

Loose MOS integrated circuits, or any product containing CMOS integrated circuits, must be protected from electro-static discharge during handling and shipping. Use conductive foam pads or conductive plastic bags, and never place MOS or CMOS circuitry in contact with Styrofoam materials.

5. Ship prepaid and insured to:

   Pro-Log Corporation
   2560 Garden Road
   Monterey, California 93940

   Reference CRO
   #_____

International Customers

Equipment repair is handled by your Pro-Log distributor. If you need to contact Pro-Log, the factory can be reached at any time by TWX at 910-360-7082.

Limited Warranty

Limited Warranty: Seller warrants that the articles furnished hereunder are free from defects in material and workmanship and perform to applicable, published Pro-Log specifications for five years from date of shipment on STD BUS Series 7000 cards and two years for PROM Programmers. This warranty is in lieu of any other warranty expressed or implied. In no event will Seller be liable for special or consequential damages as a result of any alleged breach of this warranty provision. The liability of Seller hereunder shall be limited to replacing or repairing, at its option, any defective units which are returned F.O.B. Seller's plant. Equipment or parts which have been subject to abuse, misuse, accident, alteration, neglect, unauthorized repair or installation are not covered by warranty. Seller shall have the right of final determination as to the existence and cause of defect. As to items repaired or replaced, the warranty shall continue in effect for the remainder of the warranty period, or for ninety (90) days following date of shipment by seller of the repaired or replaced part whichever period is longer. No liability is assumed for expendable items such as lamps and fuses. No warranty is made with respect to custom equipment or products produced to Buyer's specifications except as specifically stated in writing by Seller and contained in the contract.

Cut Here

FOLD

FOLD

Cut Here

PRO-LOG® CORPORATION

Yes, I'm interested in Pro-Log.

Please send me more information about your:

- ☐ 7000 Series STD BUS Cards
- ☐ STD BUS Packaging Solutions
- ☐ STD BUS Subsystem
- ☐ STD BUS Systems
- ☐ Designer Software
- ☐ Design Courses and Seminars
- ☐ Programmer Product Line
- ☐ Other _____

- ☐ I am currently using Pro-Log equipment.
- ☐ Add my name to your mailing list.
- ☐ Please have a Sales Representative contact me.

Reader Comments:

Name _____ City _____

Company _____ State _____ Zip _____

Address _____ Telephone _____

PLACE
POSTAGE
HERE

PRO-LOG®
CORPORATION
2560 GARDEN ROAD
MONTEREY, CA 93940

Attn: Literature Department

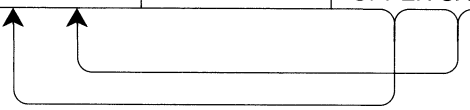
NOTES

NOTES

NOTES

NOTES

HEX				MSD	p=1	8	9	A	B	C	D	E	F
					p=0	0	1	2	3	4	5	6	7
	BITS				b8	p	p	p	p	p	p	p	p
					b7	0	0	0	0	1	1	1	1
					b6	0	0	1	1	0	0	1	1
LSD	b4	b3	b2	b1 \ b5		0	1	0	1	0	1	0	1
0	0	0	0	0		NUL	DLE	SP	0	@	P	‘	p
1	0	0	0	1		SOH	DC1	!	1	A	Q	a	q
2	0	0	1	0		STX	DC2	"	2	B	R	b	r
3	0	0	1	1		ETX	DC3	#	3	C	S	c	s
4	0	1	0	0		EOT	DC4	$	4	D	T	d	t
5	0	1	0	1		ENQ	NAK	%	5	E	U	e	u
6	0	1	1	0		ACK	SYN	&	6	F	V	f	v
7	0	1	1	1		BEL	ETB	’	7	G	W	g	w
8	1	0	0	0		BS	CAN	(	8	H	X	h	x
9	1	0	0	1		HT	EM	)	9	I	Y	i	y
A	1	0	1	0		LF	SUB	*	:	J	Z	j	z
B	1	0	1	1		VT	ESC	+	;	K	[	k	{
C	1	1	0	0		FF	FS	,	<	L	\	l	|
D	1	1	0	1		CR	GS	–	=	M	]	m	}
E	1	1	1	0		SO	RS	.	>	N	^	n	~
F	1	1	1	1		SI	US	/	?	O	_	o	DEL
						CONTROL				UPPER CASE		LOWER CASE	



CONTROL CHARACTERS AND STANDARD KEYBOARD CONTROL SHIFT CHARACTERS

CTRL	@	NUL	Null	CTRL	K	VT	Vertical Tabulation	CTRL	V	SYN	Synchronous Idle
CTRL	A	SOH	Start of Heading	CTRL	L	FF	Form Feed	CTRL	W	ETB	End of Transmission Block
CTRL	B	STX	Start of Text	CTRL	M	CR	Carriage Return	CTRL	X	CAN	Cancel
CTRL	C	ETX	End of Text	CTRL	N	SO	Shift Out	CTRL	Y	EM	End of Medium
CTRL	D	EOT	End of Transmission	CTRL	O	SI	Shift In	CTRL	Z	SUB	Substitute
CTRL	E	ENQ	Enquiry	CTRL	P	DLE	Data Link Escape	CTRL	[	ESC	Escape
CTRL	F	ACK	Acknowledge	CTRL	Q	DC1	Device Control 1	CTRL	\	FS	File Separator
CTRL	G	BEL	Bell	CTRL	R	DC2	Device Control 2	CTRL	]	GS	Group Separator
CTRL	H	BS	Backspace	CTRL	S	DC3	Device Control 3	CTRL	^	RS	Record Separator
CTRL	I	HT	Horizontal Tabulation	CTRL	T	DC4	Device Control 4	CTRL	_	US	Unit Separator
CTRL	J	LF	Line Feed	CTRL	U	NAK	Negative Acknowledge			DEL	Delete

NOTE: The table above lists the keyboard character to be pressed with the Control key, the mnemonic for the function to be performed, and the description of the function. While the keyboard character is listed as uppercase, either uppercase or lowercase characters may be used.

ASCII Code Assignments

USER'S MANUAL

PRO-LOG® CORPORATION
2560 Garden Road, Monterey, California 93940
Phone: (408) 372-4593, (800) 538-9570
Telex: 171879, TWX: 910-360-7082

111700-002C 500 7/86